OPTIMIZATION OF SPLINE INTERPOLATED ROBOT JOINT TRAJECTORIES BY THE METHOD OF LOCAL VARIATIONS

by

SUSHMA SINHA

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

FEBRUARY 1990

OPTIMIZATION OF SPLINE INTERPOLATED ROBOT JOINT TRAJECTORIES BY THE METHOD OF LOCAL VARIATIONS

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

SUSHMA SINHA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

FEBRUARY 1990

EE-1990-M-SIN-OPT

CERTIFICATE

Certified that the work presented in this thesis entitled 'OPTIMIZATION OF SPLINE INTERPOLATED ROBOT JOINT TRAJECTORIES BY THE METHOD OF LOCAL VARIATIONS' has been carried out by Sushma Sinha under my supervision and the same has not been submitted elsewhere for a degree.

S. S. Prabhu.

(Dr. S. S. Prabhu)

Professor

Department of Electrical Engineering

Indian Institute of Technology

KANPUR

To

Dr. Prabhu

ABSTRACT

A combination of B-Spline representation and the metho of local variations (MLV) has been used for off-line optima planning of robot joint trajectories. B-Splines have severa advantages, for robot trajectory representation, in terms o accuracy of the representation and its derivatives and in terms o the ease with which computations can be done. Furthermore, becaus of their property of local support, B-Splines are well-suited fo use with MLV. Two types of optimal joint trajectories have bee determined: (1) trajectories to obtain minimum energy loss in th joint motors, for fixed time of traversal of the end effecto between two given points in cartesian space and (2) trajectorie which minimize the time of traversal of the end effector betwee two given points in cartesian space. The trajectories obtaine satisfy constraints on position, velocity, acceleration and jer in the joint space. The algorithms developed have been used t solve the planning problems posed for PUMA - 560 robo manipulator. Though obstacle constraints have not been considere here, they can be incorporated without difficulty in th algorithms developed .

CONTENTS

LIST OF SYMBOLS

K = Order of the B-Spline

k = Discrete time instant

η = A knot point of the original knot sequence

ζ = A knot point of the extended knot sequence

$\mathcal{P}_{K,\eta,K-2}$ = Space of spline functions of order K, with the continuous first K-2 derivatives over the knot sequence $\{\eta_i = i = 1, 2, . . . ,m+1\}$.

B_i^K = ith B-Spline of order K

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The need for flexibility, improved productivity, better quality and lower production cost in industries has motivated the speedy growth of automation in manufacturing processes. Special purpose machines were the first step towards this automation. Development of NC and CNC machine tools was the next major event in autoamation in manufacturing processes. The developments which are now taking place in Robotics will ultimately usher in an era of automatic factories.

An industrial robot is a multifunctional computer-controlled mechanical manipulator capable of performing a variety of tasks through motions along several directions. Robot manipulators need to be controlled in efficient and effective ways to enable them to perform the multitude of complex tasks for which they are designed. So, Robot control has evolved as an important R & D area for control engineers.

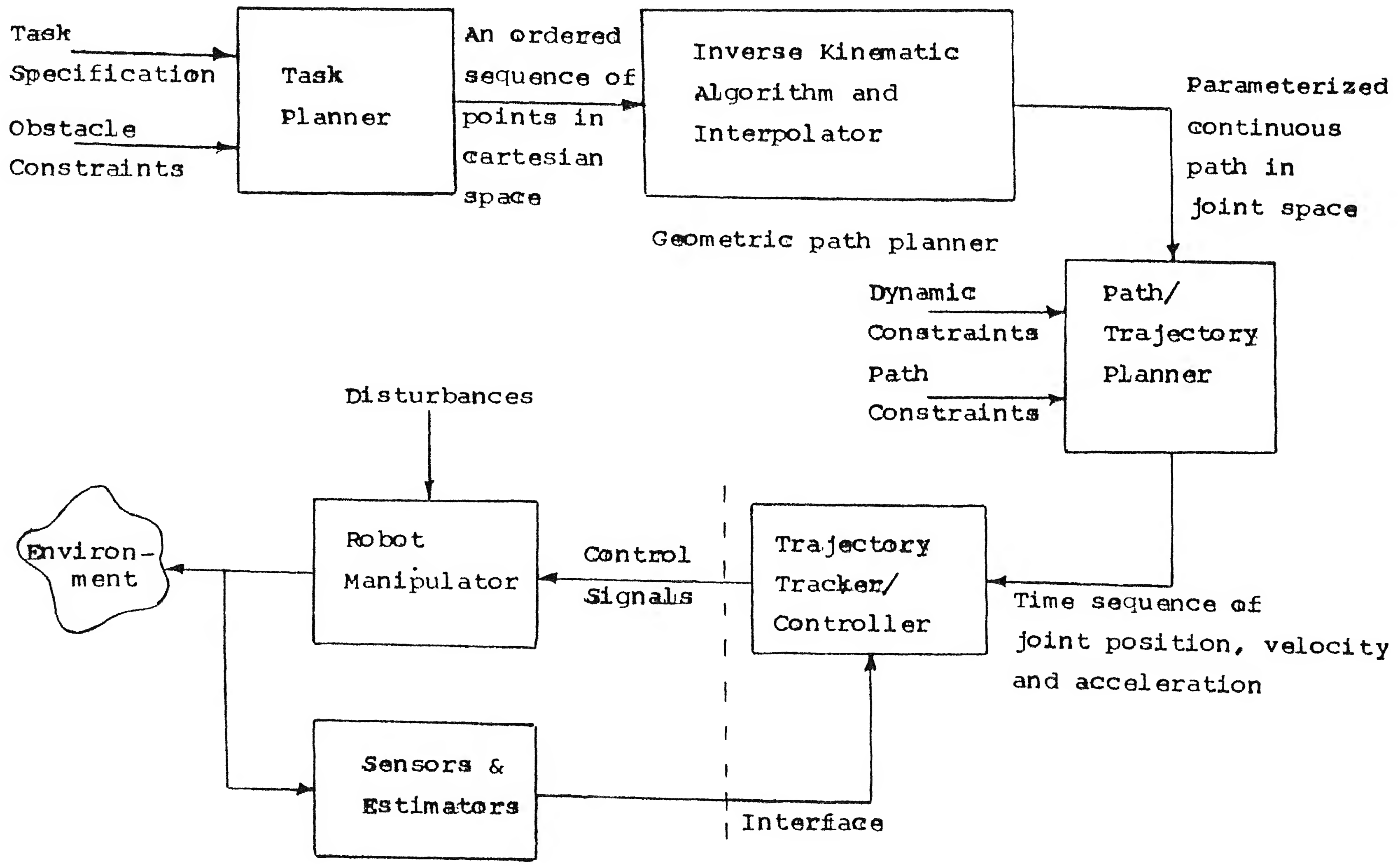
1.1 Trajectory Planning And Motion Control

Each of the robot joints is provided with its own control loop which is actually a servomechanism. Actuators driving the robot joints are generally dc servomotors. So, the control of robot manipulator is actually the control of joint actuators to make the end effector (EE) move efficiently along a prespecified path. Efficiency here implies mainly two types of control, namely, control for minimum energy and minimum time operation. In the former type of control, robot manipulators are operated in such a way that the energy required to move the EE from an initial position to the target position is minimum and in the latter type the EE moves from the initial to the final location in minimum time. The basic problem here is to determine the appropriate control signals to the joint servomechanisms.

Because of the nonlinearity and coupling in robot dynamics, the control problem of robot manipulator is much involved and requires large computational effort for its solution. Usually trajectory control is done in two steps : Off-line optimal trajectory planning and On-line tracking. A schematic diagram of robot trajectory control is shown in Fig. 1.1 .

First of all, the desired task must be specified to Task Planner, that is, the EE position and orientation, both initial and final, must be specified in cartesian coordinates. Secondaly, Task Planner must have a complete knowledge of obstacles in cartesian coordinates. An obstacle may be static or dynamic. If it is dynamic, its position as a function of time must be known apriori. Task Planner generates an ordered sequence of points in cartesian space which results in a collision-free path.

Off-line optimal path plannig is usually done in joint space because of the direct involvement of controlled variables, fast computation time and ease of handling dynamic constraints on torques or forces. Also, it is simpler than cartesian space planning. So, Geometric Path Planner first uses the inverse kinematic algorithm (IKA) to find the values of joint variables at



Task Specification
Obstacle Constraints
Task Planner
An ordered sequence of points in cartesian space
Inverse Kinematic Algorithm and Interpolator
Geometric path planner
Parameterized continuous path in joint space
Dynamic Constraints
Path Constraints
Path/ Trajectory Planner
Time sequence of joint position, velocity and acceleration
Trajectory Tracker/ Controller
Control Signals
Disturbances
Robot Manipulator
Environ-ment
Sensors & Estimators
Interface

Fig 1.1 - A schematic diagram of robot trajectory control

knot points corresponding to the cartesian knot sequence. Cartesian knot sequence is the sequence of points at which EE positions and orientations are specified. Now, having specifiction of joint variables at knot points, continuous joint trajectories in parameterized form are obtained by a suitable interpolation algorithm.

Interpolator output is fed to Trajectory Planner. Path constraints are given as input to Trajectory Planner. Path constraints are specified in terms of maximum reach (position constraints), maximum velocity, maximum acceleration, etc. Also path constraints specify some performance criterion to be minimized, (usually time), in traversing the EE along a particular path. Here comes the optimization problem into picture, that is, the joint trajectoris should be optimized for some criterion function together with satisfaction of all the constraints (boundary constraints and path constraints). Dynamic constraints such as constraints on actuator torque/force etc are also specified and taken into account by the trajectory planner. Thus the output of the trajectory planner is the optimal time sequence of position, velocity, acceleration, etc. Here, it should be noted that the optimization algorithm does not change the geometric path, i. e. , the EE moves along the path decided by the task planner. Only the velocities and accelerations of various joints are determined at various time instants to optimize the performance criterion.

It is clear that the geometric path itself can be modified during optimization. This can substantially improve

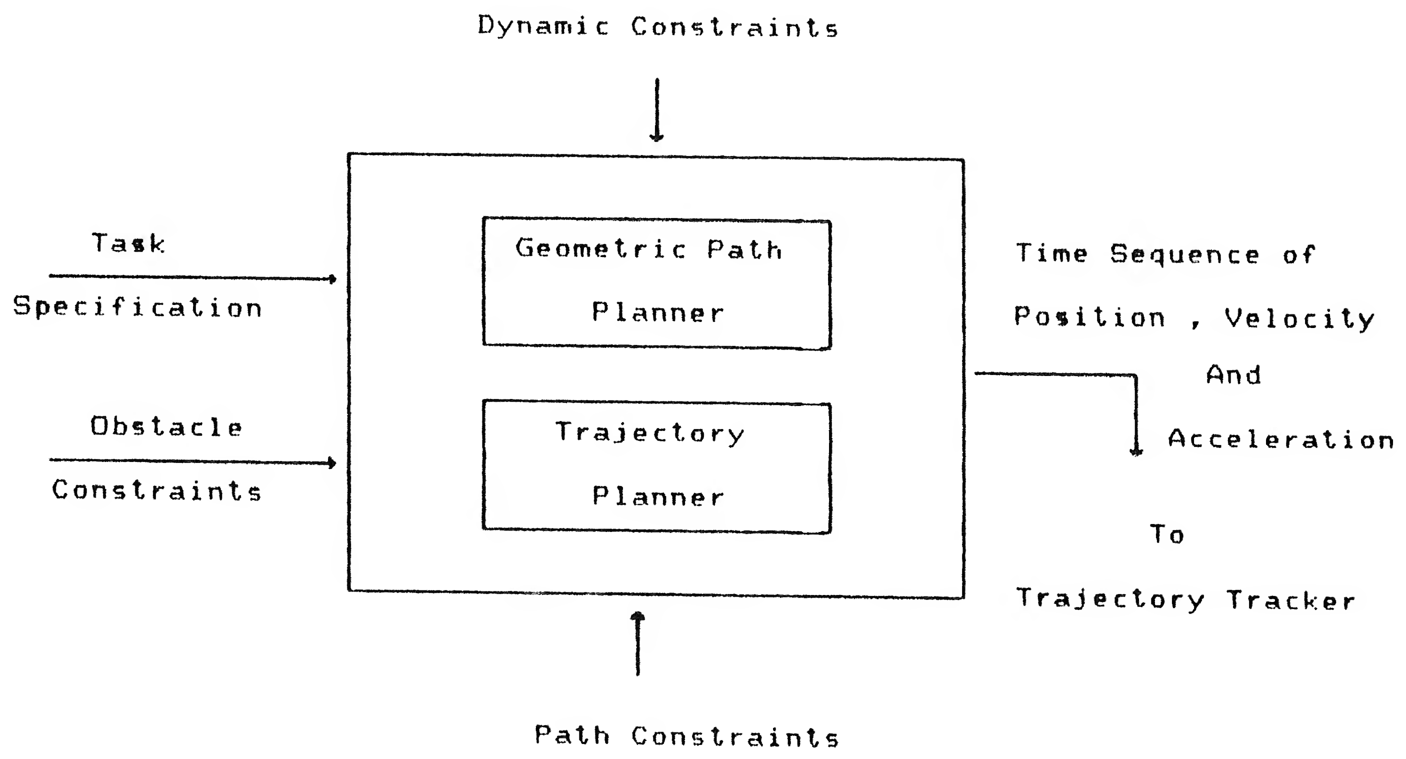


Fig. 1.2 Simultaneous Geometric Path Planning And Trajectory Planning

robot performance in comparison with that obtained by a non-optimized geometric path, [7]. For this we can have a scheme where a preliminary geometric path planning and trajectory planning exercise can be done to generate not only a geometric path but also a feasible non optimal trajectory with respect to time in cartesian space according to some rule. This cartesian trajectory is then converted into joint space trajectories. The main trajectory planner takes obstacles into account and uses the Direct Kinematic Algorithm (DKA) to find the positions of the different joints and links in cartesian space in order to check the obstacle presence. It uses a suitable optimization algorithm to obtain an optimal robot trajectory specified in terms of joint space variables. We can further consider that this trajectory planner also executes the initial task of generating a feasible trajectory. In this Trajectory Planner, the traditional distinction between Geometric Path Planner and Trajectory Planner is no longer present [Fig. 1.2].

The output of the trajectory planner is fed to the Trajectory Tracker, where the actual manipulator movement is compared with the planned one and, according to the error encountered, corrective measures are taken by the appropriate controllers.

1.2 Review Of The Earlier$ Works

Various contributions have been made in the past decade to the area of optimal path planning. A brief review of some of them is given below.

Luh and *Lin* [3] have solved the Minimum Time Trajectory Planning (MTTP) problem by using nonlinear programming. The path to be optimized was defined by a set of corner points connected through straight lines with circular arcs in the neighbourhood of corner points. Constant or piecewise constant bounds with respect to time were assumed on robot velocity and acceleration. These bounds depend on position, payload mass and its shape etc. Thus, for efficient and safe working of this scheme, the upper bounds were necessarily chosen as the global greatest lower bounds of velocity and acceleration values, that is, the worst case limits were used. This implies underutilization of robot capabilities and may result in considerable inefficiency. For example, robot may take too much time to perform a particular task.

Shin and *McKay* [4] and *Bobrow* [5] attempted solving MTTP problem using phase plane technique. Robot dynamics was described using parametric function in joint space. Geometric path constraints and dynamic constraints were taken into account. Both the methods assume that there is a set of admissible velocities for a given position which leads to the concept of an admissible region in the phase plane. The solution to the optimization problem has been obtained by a switching curve in the phase plane. The method of [4] is applicable to the general case of non-simply connected admissible regions while [5] works on the assumption of simply-connected admissible regions.

Kin and *Shin* [6] presented a solution to MTTP problem in joint space. The path was assumed to be straight line between corner points. The solution to the global MTTP problem was obtained through local optimization. For this, the actual problem was replaced by a set of local optimization problems, one at every local corner point. This method is not applicable for the case where a path segment is too short to include the cruise stage.

Shin and *McKay* [7] have again taken up the MTTP problem. In this work, first a lower bound was derived on the time required for a particular manipulator movement and then this bound has been minimized. The paper, thus has developed a method for finding a geometric path for robot motion from a given initial point to a given final point in minimum time. Parameterization approach has been used to describe the geometric path.

Suh and *Shin* [8] presented a near optimal path for the EE movement using variational calculus and dynamic programming methods. The cost function was defined as a weighted distance function so that obstacles could be avoided and the total distance of travel minimized simultaneously.

A totally discrete approach has been adopted by *Tan* and *Potts* [9] for the solution of MTTP problem. This uses the method of approximate programming (MAP), which is a sequence of linear subprograms, to solve this problem.

A technique for solving MTTP with obstacle constraints has been presented by *Bobrow* [10] through the use of a distance function. A B-Spline polynomial representation has been used for the cartesian path of the manipulator.

The method of local variations (MLV) has been used by *Patrikar* [11] and *Seshadri* [12] to solve Minimum Energy Trajectory

Planning (METP) and MTTP problems. [12] has taken obstacles, both static and dynamic, into consideration. Finite difference method has been used for the approximation of joint trajectories in both the cases.

Lin and *Luh* [13] and *Patel* have solved optimal path planning problem using B-Splines. The optimization criterion is minimum time in both the cases. [13] uses nonlinear programming to solve this problem while [14] uses linear scaling. They have not taken dynamic constraints into consideration.

1.3 Objective Of The Present Work

In this thesis two problems of off-line optimal trajectory planning have been discussed, namely, minimum energy and minimum time trajectory planning. The objective of the first problem is to find an optimal EE path to move the EE from an initial specified position to the given target position in specified time, so that the electrical energy consumption by the robot drive motors is minimized. In the second problem an optimal path is found for the EE to go from a given initial position to a given final position in minimum time.

For the sake of simplicity trajectory planning has been done in a collision free environment. However, this method can take account of obstacles with equal efficiency. In both the problems B-Splines have been used for interpolation of joint trajectories.

The algorithm used in this thesis for optimization of the spline interpolated joint trajectories, called the Method of Local Variations (MLV), was originally developed by *Chernous'Ko*. It is basically a method to obtain the numerical solution of fixed time optimal control problems, by systematic local perturbations of the state trajectory. This method can take account of magnitude constraints on state variables easily.

The main contribution of the work reposed here lies in the use of a combination of B-Splines and MLV. As MLV works on the concept of systematic local perturbations of a feasible trajectory to find an optimal one, B-Spline representation of trajectory makes it easy to carry out local perturbations since B-Splines have the property of limited support. So, if a spline interpolated trajectory is perturbed by perturbing a spline coefficient, only a small portion of the whole trajectory gets perturbed, not the whole trajectory. Secondly, these splines enable us to easily incorporate derivative continuities. Thus, we can have continuous position, velocity acceleration and jerk if fifth order splines are used. This is a very desirable feature for any trajectory planning scheme.

1.4 Organization Of The Thesis

Chapter 2 presents piecewise polynomial approximation and gives a brief introduction to B-Splines.

Chapter 3 presents an interpolation scheme using B-Splines. It also describes the essential features of the method of local variations (MLV). Pseudocodes for the interpolation scheme and optimization using MLV in combination with B-Splines are given.

Chapter 4 considers the METP problem using both cubic and quartic B-Splines. Results of computer simulation of the first three joints of PUMA 560 robot are given.

Chapter 5 gives a solution to the MTTP problem. An algorithm using MLV and B-Splines is given. Computer simulation results are presented.

Conclusions of the present study are presented and discussed in chapter 6. Some suggestions for future work in this area are also given.

CHAPTER 2

AN INTRODUCTION TO B-SPLINES

Given a function $f(x)$ over an interval $[a,b]$, we say that another function $\psi(x)$ is an approximation to it, if $\psi(x)$ is close to $f(x)$ in some sense. Closeness of $f(x)$ and $\psi(x)$ can, for example, be judged in terms of a suitable norm. Thus, if $f(x)$ and $\psi(x)$ are members of a normed linear space, then the closeness of the functions can be judged in terms of the norm of $f(x) - \psi(x)$, represented by the notation $\| f - \psi \|$. For these functions to be close, the above norm should be small. One way of obtaining an approximation to a function $f(x)$ over an interval $[a,b]$ is to utilize values $f(x_i)$ of the function at a discrete set of points, $x_i \in [a,b]$, $i = 1 ,2 , ... ,n$. If the approximation $\psi(x)$ is such that $f(x_i) = \psi(x_i)$, $i = 1 , 2, ... , n$, then we say that $\psi(x)$ is an interpolating approximation to $f(x)$ or simply as interpolating function for the data $f(x_i)$, $i = 1, 2, ... ,n$.

Polynomials are widely used as interpolating functions . The main advantage in their use is the ease with which they can be evaluated , differentiated and integrated with the aid of simple arithmetic operations . A polynomial of order n can be represented as

$$p(x) = a_0 + a_1 x + . . . + a_n x^{n-1} \qquad (2.1)$$

We will only consider real valued functions of real variables. Let $\mathcal{P}_n$ represent the set of all polynomials of order n. Then with the usual rules of addition and scalar multiplication, this set forms a linear space.

Some disadvantages with polynomial interpolation are the following :

In order to have good accuracy, it is often necessary to have high degree polynomials and to determine the large number of coefficients involved we have to use sufficiently large number of data points. There is usually large computational effort,and numerical problems are encountered in the determination of the coefficients.

Polynomials yield good approximation if interpolation points are chosen appropriately and the function to be approximated behaves in a uniform manner throughout the interval of approximation. In fact, the polynomial interpolant is very sensitive to the choice of interpolation points [1]. Furthermore , if the behaviour of the function to be approximated differs significantly over a subinterval of the interval of approximation from its behaviour elsewhere, (for example rapid oscillations over a subinterval), then the approximation is poor everywhere [1]. This global dependence of approximation over local properties of the function to be approximated makes polynomial approximation unattractive in many situations. Piecewise polynomial approximations are much better suited for these situations. B-Spline approximations are a special form of piecewise polynomial approximations. These approximations yield better accuracy and have advantages in numerical computations. We will develop the concept of these approximations below.

2.1 PIECEWISE POLYNOMIAL (PP) FUNCTION

Let $\langle \eta_i \rangle$, $i = 1, 2, \ldots, m+1$ be a strictly increasing sequence of points, that is, let η_i satisfy the condition $a = \eta_1 < \eta_2 < \ldots < \eta_m < \eta_{m+1} = b$. Let K be a positive integer. $s(x)$ is defined to be a pp function of order K (degree $\le$ K-1) , over the interval $[a,b]$, if it is a polynomial of order K (degree $\le$ K-1) in each of the intervals $[\eta_i, \eta_{i+1}]$, $i = 1, 2, \ldots, m$.

2.2 SPLINE FUNCTIONS

Actually spline is a mechanical device used by draftsmen for drawing a smooth curve. It consists of a strip of rod of some flexible material which can be bent to constrain it to approximately pass through certain plotted points on a graph. The term spline function implies that the graph of such a funtion is similar to a curve drawn by a mechanical spline. Spline functions are a class of pp functions satisfying the additional property of derivative continuity, as explained below

Definition - A spline function $s(x)$ of order K is defined to be continuous pp of order K if following two conditions are satisfied

(i) it is in $C[a,b]$ (i.e, $s(x)$ is a continuous function of x over the interval $a \le x \le b$), and there exists a strictly increasing sequence of real points $a = \eta_1 < \eta_2 < \ldots < \eta_{m+1} = b$ such that $s(x)$ is a polynomial of degree at most K-1 on each of the intervals $[\eta_{i-1}, \eta_i]$, $i = 2, 3, \ldots, m+1$.

(ii) it is also in the space $C^{K-2}[a,b]$, *i.e.*, it has continuous

derivatives up to (K-2) th order over the interval $a \le x \le b$. The points η_i, i = 1,2, . . . ,m+1 are called knot points or simply as knots.

2.3 Linear Space Of Spline Functions

If $s_1(x)$ and $s_2(x)$ are spline functions of order K over [a,b] defined with respect to the knot sequence $\langle\eta_i\rangle$, i = 1, 2, . . . ,m+1, then $\alpha\ s_1(x) + \beta\ s_2(x)$ is also a spline function of order K with respect to the same knots, where α and β are any two scalars. Thus we can speak of the linear space of spline functions of order K over [a,b] with respect to the knots η_i, i = 1, . . . , m+1.

Now , if we consider pp functions defined over m+1 knots η_1, η_2, . . . ,η_{m+1} and if over each subinterval $[\eta_i, \eta_{i+1}]$ the pp function is represented in terms of a Kth order polynomial (characterized in terms of K parameters), then, in all we require Km parameters for the representation of the function. Thus the dimension of such a space of pp functions is Km. This space is represented by the symbol $\mathcal{P}_{K,\eta}$. Now, if we impose continuity condition to be satisfied at the knots, the number of free parameters to define the function is is Km - (m-1). If the pp function is to have continuous derivatives up to order K-2, then the number of free parameters required to define the function is Km - (m-1)(K-1) = m+K-1. Thus, the dimension of the space of spline functions of order K defined with respect to knots η_i, i = 1 , 2 , . . . , m+1 is (m+K-1). If the pp function is to have continuous derivatives at the knot points up to order ν , $\nu \le K-2$, then the number of free parameters, or alternatively, the dimension of the space of these functions is $Km - (m-1)(\nu+1)$. We will use the symbol $\mathcal{P}_{K,\eta,\nu}$ to denote this space. Clearly, $\mathcal{P}_{K,\eta\ \nu}$ is a subspace of $\mathcal{P}_{K,\eta}$. In this nc.ation, the space of spline functions is denoted by $\mathcal{P}_{K,\eta,K-2}$.

2.4 Selection Of A Basis For The Space Of Splines , $\mathcal{P}_{K,\eta,K-2}$.

Computational efficiency for problems employing splines requires construction of a suitable basis for the space of splines $\mathcal{P}_{K,\eta,K-2}$, that is , we need a sequence ϕ_1, ϕ_2, ... , of functions , all in $\mathcal{P}_{K,\eta,K-2}$ such that every element of $\mathcal{P}_{K,\eta,K-2}$ can be written as a unique linear combination , $\sum_j \alpha_j\ \phi_j$, of the sequence ϕ , ϕ , ... , .

2.4.1 The Truncated Power basis .

Each member of the subspace $\mathcal{P}_{K,\eta,K-2}$ can be written as

$$f(x) = \sum_{j=0}^{K-1} c_j \quad x^j + \frac{1}{(K-1)\ !} \sum_{j=2}^{m} d_j \quad (x-\eta_j)_+^{K-1} \quad ,a \le x \le b \qquad (2.2)$$

where

$$(x-\eta_j)_+ \equiv \max\ [0,\ x-\eta_j]$$

The functions of the type $(x-\eta_j)_+$ are called truncated functions. The parameters C_j, $j = 0,1,\ .\ .\ .\ ,K-1$ and d_j , $j = 2,\ .\ .\ .\ ,m$ distinguish the different members of $\mathcal{P}_{K,\eta,K-2}$. Clearly $f(x)$ is continuous and the derivatives are continuous up to $\frac{d^{K-2}}{dx^{K-2}} f(x)$. After this if we differentiate once again $\frac{d^{K-1}}{dx^{K-1}} f(x)$ will not in general be continuous.

Since the space $\mathcal{P}_{K,\eta,K-2}$ is of dimension m+K-1, we may choose any convenient (m+K-1) number of basis functions. Let $\phi_j(x)$, j = 1, 2, . . . ,m+K-1 be one such basis. Then

$$f(x) = \sum_{j=1}^{m+K-1} \lambda_j \, \phi_j(x) \quad , \qquad a \le x \le b$$

Clearly we can choose the basis function as

$$\phi_j(x) = (x-\eta_j)_+^{K-1} \quad , \quad a \le x \le b \quad , \; j = 2, 3, \; . \; . \; . \; ,m$$
$$= x^{j-(m+1)} \quad , a \le x \le b \; , \quad j = m+1, \; m+2, \; . \; . \; . \; ,m+K$$

This follows directly from the representation we gave earlier for f(x).

The above basis which uses truncated power functions is, however, not convenient, especially in numerical computations, because of two reasons :

(i) At a point x, f(x) involves in general more than K coefficients.

(ii) For very non-uniform set of knot points $\{\eta_i\}$, some of the basis functions ϕ_j become nearly linearly dependent on the others. This will cause problems in numerical work.

2.4.2 The B-Spline Basis

The so called B-Spline basis overcomes both the objections given above to truncated power basis. The need, essentially, is to choose basis functions which are identically zero over a large part of the interval of approximation, $a = \eta_1 \leq x \leq \eta_{m+1} = b$; in other words, the need is to choose functions which have limited support over the interval. Thus the problem is to find an element of $\mathcal{P}_{K,\eta,K-2}$ that is identically zero over two intervals $[\eta_1, \eta_p]$ and $[\eta_q, \eta_{m+1}]$, but is not identically zero on (η_p, η_q). Such a function can be represented as

$$f(x) = \sum_{j=p}^{q} d_j \, (x-\eta_j)_+^{K-1} \qquad , a \leq x \leq b \qquad (2.4)$$

with the additional condition

$$\sum_{j=p}^{q} d_j \, (x-\eta_j)_+^{K-1} \equiv 0 \qquad \text{for } \eta_q \leq x \leq b \qquad (2.5)$$

Thus, over the int rval $p < x < q$, $f(x)$ has a Kth order polynomial representation and outside this interval, it is identically zero. Equating the sum of the coefficients of like powers of x to zero in the expansion of the above expression, we get

$$\sum_{j=p}^{q} d_j \, \eta_j^i = 0 \qquad , i = 0, 1, \ldots, K-1 \qquad (2.6)$$

These equations are homogeneous and have a non - trivial solution for d_j's only if the number of equations is less than the number of unknowns, that is, only if $(q-p) \geq K$, because, then the number of coefficients d_j, $(j = p, p+1, \ldots, q)$, is greater than the number of equations, which is K.

If $q-p = K$, it can be shown [2] that

$$d_j = \prod_{\substack{i=p \\ i\neq j}}^{p+K} \frac{1}{(\eta_i - \eta_j)} \qquad , (j = p, p+1, \ldots, p+K) \quad (2.7)$$

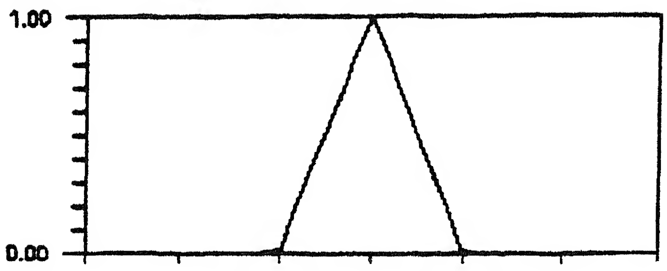


Fig. 2.1 B–Spline of order 2.

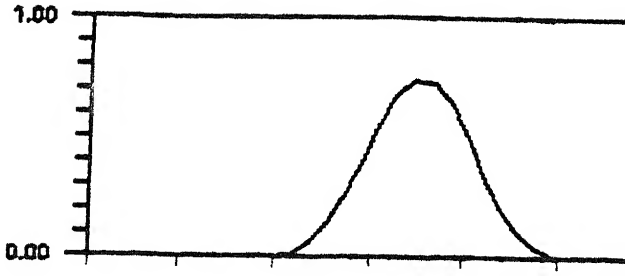


Fig. 2.2 B–Spline of order 3

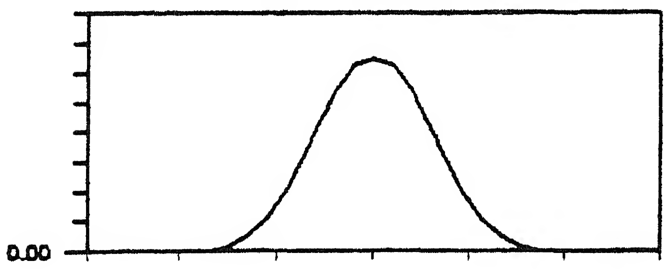


Fig. 2.3 B–Spline of order 4

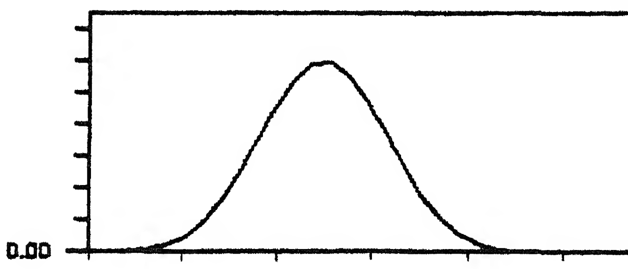


Fig. 2.4 B–Spline of order 5

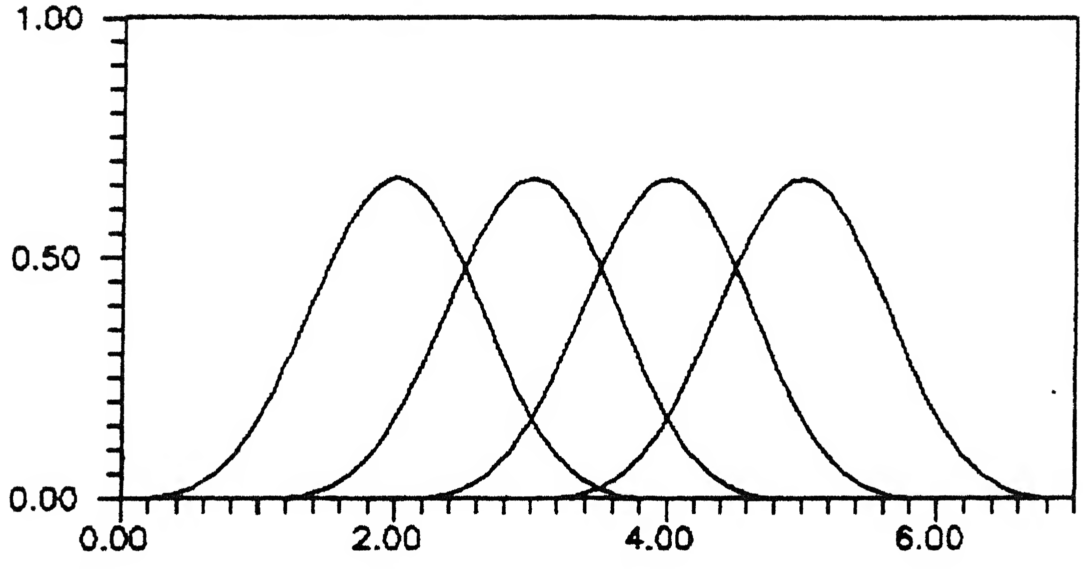


Fig. 2.5 – Figure showing the linearly independent spline functions of order 4 over 8 knots.

The spline function

$$B_p^K(x) = \sum_{j=p}^{p+K} \left[\prod_{\substack{i=p \\ i\neq j}}^{p+K} \frac{1}{(\eta_i - \eta_j)} \right] (x-\eta_j)_+^{K-1} \quad , -\alpha < x < \alpha \tag{2.8}$$

is called a B-Spline The subscript p denotes that $B_p^K(x)$ is non-zero only if x is in the interval (η_p, η_{p+K}). Figures (2.1) to (2.4) show B-Splines of degree one, two, three and four respectively for the case with uniform knot spacing.

The property of limited support of a basis function is very useful in computer calculations. Therefore we seek a basis of $\mathcal{P}_{K,\eta,K-2}$ that is composed of B-splines. The functions $B_p^K(x)$, (p = 1, 2, . . . ,m-K+1) are linearly independent, [Fig. 2.5] and are all in $\mathcal{P}_{K,\eta,K-2}$. Now the dimension of the space spanned by these (m-K+1) functions is (m-K+1) while the dimension of $\mathcal{P}_{K,\eta,K-2}$ is m+K-1. Therefore, another 2K-2 basis functions are required. A convenient way of choosing them is to introduce some extra knots outside the interval [a,b]. We select these additional knots on the real line as follows :

$$\zeta_1 \le \zeta_2 \le . . . \le \zeta_{K-1} \le \eta_1 = a$$

$$b = \eta_{m+1} \le \zeta_{m+K+1} \le . . . \le \zeta_{m+2K-1}$$

The original and additional knot points form the extended knot set, ζ_j,(j = 1, 2, . . . ,m+2K-1). Though the B-Splines are defined with respect to this extended knot set, we will use function values only when $x \in [a, b]$. Thus the total number of B-Splines has been made equal to the dimension of $\mathcal{P}_{K,\eta,K-2}$ and any $f(x) \in \mathcal{P}_{K,\eta,K-2}$ can be expressed as

$$f(x) = \sum_{j=1}^{m+K-1} \lambda_j \, B_j^K (x) \quad , \quad x \in [a, b] \qquad (2.9)$$

2.5 A Recurrence Relation For B-Splines

In computer calculations, we need to frequently determine the values of B-Splines, given the value of x. We can calculate the value directly from the expression (2.8). This however is not convenient, especially when x is close to ζ_{p+K} [2]. A procedure that is efficient in all cases is described below. We will state a theorem without giving proof [2]

Theorem - Let K be an integer > 2 and let ζ_j , j = p, p+1, . . . , p+K be a set of real, strictly increasing numbers. Then $B_p^K(x)$ is given by

$$B_p^{K-1} = \frac{(x - \zeta_p) \, B_p^{K-2}(x) + (\zeta_{p+K} - x) \, B_{p+1}^{K-2}(x)}{(\zeta_{p+K} - \zeta_p)} \qquad (2.10)$$

for all real values of x.

A convenient way of calculating $B_p^i(x)$ for a given x is to compute the columns of Table 2.1, in sequence.

Table 2.1

$$\begin{tikzcd}
B_p^2(x) \arrow[r] & B_p^3(x), \ldots\ldots, & B_p^{K-1}(x) \arrow[r] & B_p^K(x) \\
B_{p+1}^2(x) = \arrow[r] & B_{p+1}^3(x), \ldots\ldots, & B_{p+1}^{K-1}(x) & \\
B_{p+2}^2(x), \ldots\ldots\ldots, & B_{p+2}^{K-2}(x) \arrow[ur, dash] & & \\
B_{p+K-2}^2(x) & \text{start here} \arrow[ul] & &
\end{tikzcd}$$

If $x \in [\zeta_i, \zeta_{i+1}]$, then the numbers in the first column have the values

$$B_j^2(x) = 0 \quad \text{for} \quad j \neq i-1 \quad , j \neq i$$

$$B_{i-1}^2(x) = \frac{(\zeta_{i+1} - x)}{[(\zeta_{i+1} - \zeta_{i-1})(\zeta_{i+1} - \zeta_i)]}$$

$$B_i^2(x) = \frac{(x - \zeta_i)}{[(\zeta_{i+1} - \zeta_i)(\zeta_{i+2} - \zeta_i)]}$$

In computer calculations, we generally use normalized B-splines for the sake of convenience By definition, normalized B-Splines are those whose sum at a given x within the specified interval is equal to 1. The recurrence relation for them given by *De Boor* [1] is as follows

$$B_i^K(x) = \frac{(x - \zeta_i)\, B_i^{K-1}(x)}{(\zeta_{i+K-1} - \zeta_i)} + \frac{(\zeta_{i+K} - x)\, B_{i+1}^{K-1}}{(\zeta_{i+K} - \zeta_{i+1})}$$

starting with (2.

$$B_i^1(x) = 1 \qquad \zeta_i \le x \le \zeta_{i+1}$$
$$= 0 \qquad \text{Otherwise.}$$

CHAPTER 3

B-SPLINES AND THE METHOD OF LOCAL VARIATIONS

Interpolation is the process of generating a continuous trajectory or curve to pass through specified data points. For this process, we need some kind of interpolating function. As explained in chapter 2, B-Spline has attractive features for interpolation. This chapter explains interpolation using B-Splines. Some numerical examples are given to show the advantages of this scheme of interpolation. In addition a brief introduction to the method of local variations for numerical solution of optimal control problems is given. This method is used later in this thesis for solving the optimal trajectory planning problem At the end we present arguments to show that B-Splines are very well suited for obtaining numerical solution of optimal control problems by the method of local variations .

3.1 Trajectory Interpolation using B-Splines

Let x_i , $i = 1, 2, . . .$, represent data points. These are the points in the interpolation interval where exact values of the function being approximated are available. These may be the result of some observations, for example, position values of a robot joint at some time instants within the traversal time .

Now consider approximation of a function $F(x)$ using normalized B-Splines of order K over m+1 knots on the interval

$[x_1, x_{m+K-1}]$ of data points . Here , $a = x_1 < x_2 < \ldots < x_{m+K-1} = b$. Note that there is no need for any of the data points to be knot points though, of course, data and knot points can coincide. The function $F(x)$ is given an approximate representation $f(x)$ as a linear combination of B-Splines over the interval $[x_1, x_{m+K-1}]$, that is ,

$$F(x) \approx f(x) = \sum_{i=1}^{m+K-1} a_i \; B_i^K(x) \qquad , \qquad x \in [x_1, x_{m+K-1}] \qquad (3.2)$$

In order to express $f(x)$ as a linear combination of m+K-1 B-Splines we need additional 2K-2 knots as was explained in chapter2. So, an extended knot set should be formed, with m+2K-1 knots.

Now , the problem is to determine the coefficients a_i, $i = 1, 2, \ldots, m+K-1$, so that the value of the function $f(x)$ can be determined at any point within the specified interval. We have the exact values of the function at the data points $(x_1, \ldots, x_{m+K-1})$. The coefficients a_i , $i = 1, \ldots, m+K-1$, can be evaluated by equating the summation in eqn. (3.2) to the exact values of the function at the data points , that is ,

$$\sum_{i=1}^{m+K-1} a_i \; B_i^K(x_j) = f(x_j) = F(x_j) \qquad , \; j = 1, \ldots, m+K-1 \qquad (3.3)$$

Thus we can get m+K-1 linear equations which can be written in matrix form as

$$B \; \underline{a} = \underline{f} \qquad (3.4)$$

where

$$\underline{a} = [a_1 , a_2 , \ldots , a_{m+K-1}]^T$$

$$\underline{f} = [f(x_1) , \ldots , f(x_{m+K-1})]^T$$

In eqn. (3.3) values of the B-Splines are calculated from eqn. (2.11) with respect to the extended knot set $\{\zeta_1, \ldots ,\zeta_{m+2K-1}\}$ As, at a particular point, only K B-Splines contribute to the interpolation process, the matrix B is a banded sparse matrix. Eqn. (3.4) can be solved for $\underline{a}$. For existence of a unique solution for $\underline{a}$, the condition of Schoenberg - Whiteny theorem [2] should be satisfied :

Schoenberg - Whiteny Theorem -

Let the numbers $\{\zeta_j , j = 1, 2, \ldots , m+K-1\}$ be strictly in ascending order. For $i = 1, \ldots ,m+K-1$, consider B-Splines $B_i^K(x)$. Let the points $\{ x_i , i = 1, 2, \ldots ,m+K-1 \}$ also be in strict ascending order. Then, for any function values $\{ f(x_i), i = 1, 2, \ldots , m+K-1 \}$, the equations

$$\sum_{i=1}^{m+K-1} a_i \; B_i^K(x_j) = f(x_j) \quad , \quad j = 1, 2, \ldots ,m+K-1$$

have a unique solution for $\{ a_i , i = 1, 2, \ldots ,m+K-1 \}$ if and only if all the numbers $\{ B_j^K(x_j) , j = 1, 2, \ldots , m+K-1 \}$ are non-zero .

After finding the coefficients $\{a_i , i = 1, 2, \ldots , m+K-1 \}$ we can evaluate the values of the function $f(x)$ for any x in the interval $[x_1, x_{m+K-1}]$ from eqn. (3.2). Also we can determine the first (K-2) derivatives of the function from the following equations

$$f^j(x) = \sum_{i=1}^{\gamma} a_{j,i}\, B_i^{k-j}(x) \quad , \qquad (3.5)$$

where

$$\zeta_{\gamma} \le x \le \zeta_{\gamma+1}$$

$$x \in [\, x_1 ,\ x_{m+K-1}]$$

$$f^0(x) \equiv f(x)$$

$$f^j(x) \equiv \frac{d^j f}{dx^j}$$

$$a_{j,i} = \frac{(K-j)\,(a_{j-1,i} - a_{j-1,i-1})}{(\zeta_{i+K-j} - \zeta_i)} \quad , \qquad (3.6)$$

$(j+1) \le i \le (m+K-1)$ and

$a_{0,i} = a_i$, $i = 1, 2, \ldots ,m+K-1$, are the coefficients in the representation of function $f(x)$ in terms of B-Splines of order K.

3.2 Boundary Constraints

At this stage we can consider inclusion of end point equality constraints in the interpolation scheme. For a Kth order B-Spline we can include a maximum of (K-2) pairs of end constraints on the values of the first (K-2) derivatives of the function. From eqn. (3.6) it is evident that the coefficients for the jth derivative representation can be expressed in terms of the coefficients for (j-1)th derivative. So, all the end point equality constraints can be written in terms of the original coefficients, a_i , $i = 1, 2, . . . , m+K-1$, and these can be included in eqn. (3.4) itself.

For illustration, we consider the case K = 4. Here we can include a maximum of 4 constraints at the boundary points of the interval of interpolation. For example, we consider the situation where constraints only on the first derivative is stipulated. We will illustrate below how this constraint can be considered.

$$f'(x) = \sum_{\gamma-2}^{\gamma} a_{1,i} \; B_i^3(x)$$

$$= a_{1,\gamma-2} \; B^3_{\gamma-2} + a_{1,\gamma-1} \; B^3_{\gamma-1} + a_{1,\gamma} \; B^3_{\gamma}$$

$$= \frac{3(\,a_{\gamma-2} - a_{\gamma-3})}{(\,\zeta_{\gamma+1} - \zeta_{\gamma-2}\,)} \; B^3_{\gamma-2} + \frac{3(\,a_{\gamma-1} - a_{\gamma-2})}{(\,\zeta_{\gamma-2} - \zeta_{\gamma-1})} \; B^3_{\gamma-1}$$

$$+ \frac{3(\,a_{\gamma} - a_{\gamma-1})}{(\,\zeta_{\gamma+3} - \zeta_{\gamma})} \; B^3_{\gamma}$$

$$= a_{\gamma-3}\left[\frac{-3\,B^3_{\gamma-2}}{(\zeta_{\gamma+1}-\zeta_{\gamma-2})}\right] + a_{\gamma-2}\left[\frac{3\,B^3_{\gamma-2}}{(\zeta_{\gamma+1}-\zeta_{\gamma-2})} - \frac{3\,B^3_{\gamma-1}}{(\zeta_{\gamma+2}-\zeta_{\gamma-1})}\right] + a_{\gamma-1}\left[\frac{3\,B^3_{\gamma-1}}{(\zeta_{\gamma+2}-\zeta_{\gamma-1})} - \frac{3\,B^3_{\gamma}}{(\zeta_{\gamma+3}-\zeta_{\gamma})}\right] + a_{\gamma}\left[\frac{3\,B^3_{\gamma}}{(\zeta_{\gamma+3}-\zeta_{\gamma})}\right]$$

$$= A_1\,a_{\gamma-3} + B_1\,a_{\gamma-2} + C_1\,a_{\gamma-1} + D_1\,a_{\gamma} \qquad (3\ 7)$$

So, the R.H.S. of eqn. (3 7) can be equated to the desired constraint and can be included in the matrix eqn. (3.4) with A_1, B_1, . . . as the elements of matrix B Solution of eqn. (3 4) gives the coefficient $\underline{a}$ automatically satisfied

Similarly for K = 5, we can include a maximum of 6 constraints at the boundary points. Thus, for instance, end-point constraints only on first and second derivatives can be imposed as shown below

$$f^1(x) = \sum_{i=\gamma-3}^{\gamma} a_{1,i}\, B_i^4(x)$$

Expressing the coefficients $a_{1,i}$'s in terms of a_i's we get the following expression

$$f^1(x) = a_{\gamma-4}\left[\frac{-4\ B^4_{\gamma-3}}{(\zeta_{\gamma+1}-\zeta_{\gamma-3})}\right] + a_{\gamma-3}\left[\frac{4\ B^4_{\gamma-3}}{(\zeta_{\gamma+1}-\zeta_{\gamma-3})} - \frac{4\ B^4_{\gamma-2}}{(\zeta_{\gamma+2}-\zeta_{\gamma-2})}\right] + a_{\gamma-2}\left[\frac{4\ B^4_{\gamma-2}}{(\zeta_{\gamma+2}-\zeta_{\gamma-2})} - \frac{4\ B^4_{\gamma-1}}{(\zeta_{\gamma+3}-\zeta_{\gamma-1})}\right] + a_{\gamma-1}\left[\frac{4\ B^4_{\gamma-1}}{(\zeta_{\gamma+3}-\zeta_{\gamma-1})} - \frac{4\ B^4_{\gamma}}{(\zeta_{\gamma+4}-\zeta_{\gamma})}\right] + a_{\gamma}\left[\frac{4\ B^4_{\gamma}}{(\zeta_{\gamma+4}-\zeta_{\gamma})}\right]$$

$$=A_1\ a_{\gamma-4} + B_1\ a_{\gamma-3} + C_1\ a_{\gamma-2} + D_1\ a_{\gamma-1} + E_1\ a_{\gamma} \qquad (3.8)$$

Similarly for the second derivative we get the following equation

$$f^2(x) = A_2\ a_{\gamma-4} + B_2\ a_{\gamma-3} + C_2\ a_{\gamma-2} + D_2\ a_{\gamma-1} + E_2\ a_{\gamma} \qquad (3.9)$$

with

$$A_2 = \frac{12\ B^3_{\gamma-2}}{(\zeta_{\gamma+1}-\zeta_{\gamma-3})\ (\zeta_{\gamma+1}-\zeta_{\gamma-2})}$$

$$B_2 = \left[\left(\frac{-12\, B^3_{\gamma-2}}{(\zeta_{\gamma+1} - \zeta_{\gamma-3})\,(\zeta_{\gamma+1} - \zeta_{\gamma-2})} \right) - \frac{12}{(\zeta_{\gamma+2} - \zeta_{\gamma-2})} \left(\frac{B^3_{\gamma-2}}{(\zeta_{\gamma+1} - \zeta_{\gamma-2})} - \frac{B^3_{\gamma-1}}{(\zeta_{\gamma+2} - \zeta_{\gamma-1})} \right) \right]$$

$$C_2 = \left[\frac{12}{(\zeta_{\gamma+2} - \zeta_{\gamma-2})} \left(\frac{B^3_{\gamma-2}}{(\zeta_{\gamma+1} - \zeta_{\gamma-2})} - \frac{B^3_{\gamma-1}}{(\zeta_{\gamma+2} - \zeta_{\gamma-1})} \right) - \frac{12}{(\zeta_{\gamma+3} - \zeta_{\gamma-1})} \left(\frac{B^3_{\gamma-1}}{(\zeta_{\gamma+2} - \zeta_{\gamma-1})} - \frac{B^3_{\gamma}}{(\zeta_{\gamma+3} - \zeta_{\gamma})} \right) \right]$$

$$D_2 = \left[\frac{12}{(\zeta_{\gamma+3} - \zeta_{\gamma-1})} \left(\frac{B^3_{\gamma-1}}{(\zeta_{\gamma+2} - \zeta_{\gamma-1})} - \frac{B^3_{\gamma}}{(\zeta_{\gamma+3} - \zeta_{\gamma})} \right) - \frac{12}{(\zeta_{\gamma+4} - \zeta_{\gamma})} \left(\frac{B^3_{\gamma}}{(\zeta_{\gamma+3} - \zeta_{\gamma})} \right) \right]$$

$$E_2 = \frac{12\, B^3_{\gamma}}{(\zeta_{\gamma+4} - \zeta_{\gamma})\,(\zeta_{\gamma+3} - \zeta_{\gamma})}$$

Now constraint equations can be written using eqns. (3.8) and (3.9) for the end points and these equations can be incorporated in eqn. (3.4) , as was done earlier for the case K = 4 .

3.3 INTERPOLATION ALGORITHM

The notations used below are the same as those used for the development of the B-Spline theory .

Step 1 Select K.

Step 2 Select m+1 knot points $(\eta_1, \eta_2, \ldots, \eta_{m+1})$, according to the desired interpolation interval.

Step 3 Form the extended knot set $(\zeta_1, \ldots, \zeta_{m+2K-1})$ according to Schoenberg and Whiteny condition

Step 4 Select the range of interpolation $[x_1, x_{m+K-1}]$ satisfying Schoenberg-Whitney conditions [2] together with m+K-1 data points $x_1, x_2, \ldots, x_{m+K-1}$ in this interval

Step 5 Form the square matrix B of order (m+K-1) according to eqn.(3.3) with respect to the data points selected in Step 4 and the extended knot set formed in Step 3.

Step 6 Obtain the function values $f(x_j)$ $j = 1, \ldots, m+K-1$

Step 7 Solve the equation $B\,\underline{a} = \underline{f}$ for $\underline{a}$.

Step 8 Form the coefficients $a_{j,i}$, ($j = 1, \ldots, K-2$, $i = j+1, \ldots m+K-1$) , according to eqn (3.6)

Step 9 Determine the function values from eqn. (3.2) and its first (K-2) derivatives from eqn (3.5) for $x \in [x_1, x_{m+K-1}]$.

We now consider a few examples to illustrate the B-Spline interpolation scheme.

Example 1 - Find the B-Spline approximations to the functions 2x , and $e^{-0.1x}$ for K = 4 within the interval [0, 15.5].

Solution - We select 0, 1, 2, 3, 5, 7, 9, 11, 13, 14, 15, 16 as the knot points. Here m = 11. The spacings between knots need not be equal. Here, unequal spacing has been selected. Six (= 2K-2) fictitious knots are added to form the extended knot set :

Extended Knot Set = $\{$ -0.6, -0.4, -0.2, 0, 1, 2, 3, 5, 7, 9, 11, 13, 14, 15, 16, 16.2, 16.4, 16.6 $\}$

Here,

$$\zeta_{K-1} = -0.2 < x_1 = 0 = \eta_1$$

$$x_{m+K-1} = 15\ 5, \quad \eta_m < x_{m+K-1} < \eta_{m+1}$$

The constraints on initial and final values of first derivative are included in the 2nd and 13th equations respectively. So, the first two and last two data points are automatically fixed as 0 and 15.5 respectively. The remaining [(m+K-1) - 4] = 10 data points are selected as the knot points only for convenience. So, the data point set is

$$\{0,\ 0,\ 1,\ 2,\ 3,\ 5,\ 7,\ 9,\ 11,\ 13,\ 14,\ 15,\ 15.5,\ 15.5\}$$

Matrix B is now formed according to the above data points. Function values or the value of derivative of function (as the case may be) at these data points are taken as the input

Results are shown in Tables 3.1 and 3.2.

Example 2- Solve the above problem using K = 5.

Solution - Inputs needed for this case are given below :

Original Knot Set = $\{$0, 1, 2, 3, 5, 7, 9, 11, 13, 14, 15, 16$\}$

Extended Knot Set = $\{$-0.8, -0.6, -0.4, -0.2, 0, 1, 2, 3, 5, 7, 9, 11, 13, 14, 15, 16, 16.2, 16.4, 16.6, 16 8$\}$

Data Point Set = $\{$0, 0, 0, 1, 2, 3, 5, 7, 9, 11, 13, 14, 15.5,

$$15.5,\ 15.5\}$$

Here, constraints on initial and final values of first derivative are included in the 2nd and 13th equations respectively and those on the initial and final values of 2nd derivative are included in 3rd and 14th equations respectively.

Results are given in Tables 3.3 and 3.4.

3.4 THE METHOD OF LOCAL VARIATIONS (MLV)

The method of local variations was originally developed by *Chernous'Ko* [18] and *Chernous'Ko & Krylov* [19] in early 60's. It is a direct numerical method for the solution of fixed-time variational problems encountered in optimal control. It has the advantage that it yields a feasible suboptimal solution at any stage of the iterative process and it can handle state variable magnitude constraints easily. In this method the performance index of the problem is minimized by successively perturbing an initial feasible state trajectory. The perturbation given at each step in the iterative process to state trajectory is over a small subinterval of the problem : the trajectory outside this interval is not disturbed. By systematic perturbations the whole trajectory is gradually changed. The perturbations are such that the constraints on state and control variables are satisfied and, furthermore, the performance index value goes on decreasing. The iterative process is stopped when convergence of performance index has been achieved. This method finds only a local optimum. It is, thus, generally necessary to start from different initial

Table 3.1

Cubic Spline Interpolation Results

x	$f(x)=2x$	$f'(x)$	$f''(x)$
0.5	0.9998143828	1.9997953941	0.0001517023
4.5	8.9998927288	2.0000710248	-0.0003659993
8.5	16.99800264	2.0001339314	0.0009603793
10.5	20.9998005010	2.0009388959	-0 0008616201
14.5	28.99737379263	1.9953686431	0.0088074727

Table 3.2

Cubic Spline Interpolation Results

x	$f(x)=e^{-0.1x}$	$f'(x)$	$f''(x)$
2.5	0.7788157051	-0.077881103	0.0076871747
4.5	0.6376319678	-0.063756501	0.0063628581
8.5	0.4273637035	-.0427237594	0.0034849449
10.5	0.3499286649	-0.0349754188	0.0034849449
14.5	0.2345509467	-0.0234948679	0.0024130314

Table 3.3

Quartic Spline Interpolation Results

x	f(x)=2x	f'(x)	f''(x)	f'''(x)
2 5	5.0006631	2.00032678	-0.0054201	-0.0006434
6.5	12.9996626	2.00018405	0.0004398	-0.0003753
10 5	20.9999813	1.99998498	0.0004808	0.0007253
12.5	25.0016454	2.00163845	0.0002140	-0.0017643
14.5	29 0008205	1.99437565	-0.0052932	0.0174612

Table 3.4

Quartic Spline Interpolation Results

x	$f(x)=e^{-0.1x}$	f'(x)	f''(x)	f'''(x)
2.5	0.7789663	-0.0778780	0.00696840	-0.0010908
6.5	0.5222545	-0.0525163	0.00469011	0.0001579
10.5	0.3501344	-0.0353048	0.00300574	0.0003797
12.5	0.2863594	-0.0283326	0.00320210	-0.0013366
14.5	0.2344950	-0.0235463	0.00294968	-0.0002159

feasible trajectories for capturing the global minimum.

3.4.1 The Optimal Control Problem

Consider the continuous time controlled system described by the set of differential equations

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), \underline{u}(t), t) \qquad (3.10)$$

where

t is the independent variable and represents time,

$\underline{x} \in \mathcal{R}^n$ is the system state vector,

$\underline{u} \in \mathcal{R}^m$ is the control vector and

$\underline{f} = [f_1, f_2, \ . \ . \ ,f_n]^T$ is a vector function of $\underline{x}$, $\underline{u}$, and t

We consider t to be in the interval [0,T], with the final time instant, T, specified. The constraints on state and control variables are given by

$$\begin{aligned} \underline{x}(t) &\in G(t) \\ \underline{u}(t) &\in U(t,\underline{x}) \end{aligned} \qquad (3.11)$$

Where G and U are generally variable closed regions of $\mathcal{R}^n$ and $\mathcal{R}^m$ respectively.

The optimal control problem is to find an admissible control $\underline{u}(t)$ that causes the system (3.10) to follow an admissible trajectory $\underline{x}(t)$ and minimizes the performance index

$$J(\underline{x}) = \int_0^T g(\underline{x}(t), \underline{u}(t), t) \, at \qquad (3.12)$$

where g is a given scalar function By admissible control and state is meant those which satisfy eqn. (3 11)

3.4.2 Elementary Operation

A procedure called Elementary Operation [19] which is basic to the Method of Local Variations is described below.

Let t' and t'' be two time instants sufficiently close to each other. Let x', x'' be two sufficiently close points in state space. Let t', t'' be in [0,T] with t' < t'' and x' ∈ G(t') and x'' ∈ G(t''). A supplementary variational problem is now stated.

Supplementary Variational Problem :

Determine the control $\underline{u}(t)$, $t \in [t',t'']$ which takes the system of eqn. (3.10) from the initial state $\underline{x}'$ to the final state $\underline{x}''$, minimizing the performance index

$$\Delta J = \int_{t'}^{t''} g(\underline{x},\ \underline{u},\ t)\ dt \qquad (3.13)$$

while satisfying constraints stipulated by eqn (3.11).

By an elementary operation is meant an operation of solving the supplementary variational problem [19] Thus, the elementary operation basically consists of

(a)determining whether a control exists which takes the system from $\underline{x}'$ at t' to $\underline{x}''$ at t'', while satisfying the constraints of eqn. (3.11) and

(b)if there exists such a control, then, to find the control which renders the performance index (3.13) minimum.

For use of elementary operation in MLV, it is not necessary that the minimization stated above be done exactly -

approximate solution of the elementary operation considerably simplifies MLV in many situations.

3.4.3 The MLV Procedure

MLV repeatedly executes the elementary operation to solve the optimal control problem defined in sec. 3.4.1. The time interval over which the optimization is to be done is divided into N subintervals, not necessarily of equal length, given by $0 < t_1 < t_2 < . \quad < t_n = T$, with N sufficiently large.

An initial feasible state trajectory $\underline{x}^0(t)$ is chosen. This, in general, is not a trivial task and special search procedures may have to be developed to generate a feasible trajectory. Now, elementary operation is applied over the interval $[0,t_1]$. This operation will not disturb the state trajectory outside this interval, but will change it over $[0,t_1]$ to give an overall performance index value over the interval $[0,T]$ lower than its value before the elementary operation. Next the elementary operation is done over $[t_1,t_2]$, etc, till the whole interval $[0,T]$ is covered. This completes one iteration of MLV. We repeat the whole process as many times as required to converge to a minimum value of performance index.

It should be noted that MLV in general captures only a local minimum.

Chernous'Ko and others [17,20,21,22] have discussed ways of organizing the details in the MLV procedures, especially in the context of discrete time problem formulation.

The main advantages of MLV are that

(i) it does not require large computer storage.

(ii) we get a feasible suboptimal state trajectory anywhere we stop the iterative process.

(iii)it is easy to handle state variable magnitude constraints and

(iv) in many situations the elementary operation turns out to be computationally simple and fast.

3.4.4 SUITABILITY OF B-SPLINES FOR MLV

If MLV is used for trajectories interpolated by B-Splines, we get the following advantages :

(i) For reestablishment of the continuous trajectory the interpolation scheme using B-Splines is highly advantageous. This helps us to use the more accurate pp approximation of system rather than that based on finite differences.

(ii) If we use B-Splines for the interpolation of state trajectory we can get the continuous time approximation of state derivatives and control functions also. These trajectories can now be discretized by simply dividing the total traversal time into N subintervals. So, we can avoid the assumption of constant input between two consecutive time instants as is done in MLV applied to the discrete time, i.e. , the finite difference, formulation of the optimal control problem. In fact, a good handle is now available for continuous information of the control functions with very good accuracy.

(iii) In MLV, the B-Spline interpolated state trajectories are perturbed indirectly by the variation of interpolation coefficients ($a_{0,i}$). This has a great advantage because of the limited support of B-Splines Thus, unlike in the finite difference method, if one interpolation coefficient is varied, the state and control will be affected only over a certain subinterval. This is, indeed, what is basically wanted in MLV B-Spline approximation is thus ideally suited for MLV and computations become simpler.

The above features are important in robot path-planning involving obstacles. Suppose the robot arm meets an obstacle at some specific time instant within the traversal time. Then the robot arm trajectory can be modified in a neighbourhood of that instant only by perturbing the involved coefficient in order that the obstacle may be avoided. It is not needed to recompute the whole trajectory.

3.4.2 MLV Algorithm With B-Splines

The following notations are used :

K = Order of the B-Spline

$a_{j,i,k1}$ = k1th B-Spline coefficient for jth derivative of the ith state variable. $j = 0, 1, . . . , K-2$; $i = 1, . . . , n$; $k1 = 1, . . . , m+K-1$. $a_{0,i,k1}$ represents coefficients for the ith state trajectory.

k represents discrete time instant t_k and satisfies $0 \le k \le N$

J[l] = Performance index associated with the whole trajectory over

[0,T] after the lth iteration.

P = Saturation indicator. It actually measures the number of successful local variations P = 0 implies that none of the variations is valid in the current iteration and saturation is reached.

Delta = Amount of perturbation in a particular B-Spline coefficient in the state trajectory representation

Del_min = Minimum permissible perturbation in a B-Spline coefficient in the state trajectory representation.

Z[k] = Performance index associated with the kth subinterval $[t_{k-1},t_k]$.

$[t_b,t_e]$ = Interval affected by a particular B-Spline coefficient.

PI1 = Performance index associated with the interval $[t_b,t_e]$ before perturbation.

PI2 = Performance index associated with the interval $[t_b,t_e]$ after perturbation.

The MLV algorithm using B-Spline representation is as follows :

Step 1 : Obtain a B-Spline approximation of feasible state trajectories and their K-2 derivatives and hence obtain the coefficients $a_{j,i,k1}$; $j = 0, \ldots, K-2$; $i = 1, \ldots, n$; $k1 = 1, \ldots, m+K-1$.

Discretize the above trajectories at instants t_k, $k = 0, \ldots, N$. Calculate

$$Z[k] = \int_{t_{k-1}}^{t_k} f(\underline{x},\ \underline{u},\ t)\ dt\ = f(\underline{x}(k),\ \underline{u}(k),\ k)\ \Delta T\ ,\ k = 1, \ldots, N$$

Calculate the initial performance index by

$$J[0] = \sum_{k=1}^{N} Z[k]$$

Step 2 : Set l = 1 and set values of Delta, C

Step 3 : Set P = 0

Step 4 : Set i = 1

Step 5 · Set k1 = C_{IN} , where a_1, a_2, . . . ,$a_{C_{IN-1}}$ are the coefficients contributing to the function representation at the initial point.

Step 6 : Determine the interval $[t_b, t_e]$ affected by $a_{0,i,k1}$

Step 7 : Set a = 1

Step 8 : Calculate

$$PI1 = \sum_{k=b}^{e} Z[k]$$

Step 9 : Delta = a*Delta

$a_{0,i,k1} = a_{0,i,k1}$ + Delta

Step 10: Calculate the ith state, its derivatives and associated controls in the interval $[t_b, t_e]$ with the new values of $[a_{j,i,k1}$, j = 0, ... , K-2] and check whether

(i) the new states are admissible

(ii)the control inputs required to translate the states in the interval $[t_b, t_e]$ along the changed trajectory are admissible.

If 'Yes' then go to Step 11 else go to Step 12.

Step 11: Calculate

$$PI2 = \sum_{k=b}^{e} Z[k]$$

```
        If PI2 ≥ PI1 , then go to Step 12 else go to Step 13.
Step 12: Restore the coefficients :
        $a_{0,i,k1} = a_{0,i,k1} - Delta$
        Set a = -a
        If a = -1 then go to Step 9 else go to Step 14
Step 13· Success met
        J[l] = J[l-1] - PI1 + PI2
        P = P + 1
Step 14· k1 = k1 + 1 ,
        if k1 ≤ $C_{FN}$ , then go to Step 6. where $a_{C_{FN+1}}$, . . .
        ,$a_{m+K-1}$ are the coefficients contributing to the function
        representation at the final point.
Step 15: i = i+1 ; if i ≤ n then go to Step 5
Step 16: l = l+1 ; if P ≠ 0 then go to Step 3
Step 17: Set Delta = Delta/C
        If Delta > Del_min then go to Step 3
Step 18: Stop.
```

Different values of Delta can be chosen for coefficients for representation of different state variables.

CHAPTER 4

MINIMUM ENERGY TRAJECTORY PLANNING

In this chapter an off-line minimum energy path planning scheme has been discussed This scheme consists essentially of two stages · formulation of initial feasible joint trajectories using B-Splines and optimization of these trajectories for minimum energy using MLV.

A feasible EE trajectory in cartesian space is initially chosen to move the EE from the specified initial point to the final point in a fixed time. Then the corresponding feasible joint trajectories are obtained using the inverse kinematic algorithm Feasibility here implies that the constraints on joint position, velocity, acceleration and joint-motor torque are not violated

The second step is the optimization of the above trajectories for minimum energy using MLV. The aim of this optimization problem is to find a path for the EE movement between the two specified points, by locally perturbing the initial feasible joint trajectories so that the total input electrical energy lost as losses in the joint motors is minimized. Here, the geometric path is not specified apriori. Only the initial and final configurations are specified. So, geometric path planning and trajectory planning are done simultaneously.

4.1 Dynamic Model Of The Robot Manipulator

For computer simulation of the robot arm motion it is necessary to have a dynamic model of the manipulator, which is actually a set of mathematical equations depicting the dynamic behaviour of the manipulator, that is, the relationship between the joint torque and joint configuration, velocity and acceleration. There are mainly two conventional approaches for obtaining the dynamic model, namely, Lagrange-Euler (L.E.) and Newton-Euler (N.E.) formulations.

The L.E. formulation for the robot dynamics is given below

$$\underline{\tau}(t) = D(q(t))\ddot{\underline{q}}(t) + \underline{h}(q(t), \dot{q}(t)) + \underline{G}(q(t)) \qquad (4\ 1)$$

where

$\underline{\tau}(t)$ = an nx1 torque vector at joints i = 1, 2, . . . ,n

$= [\tau_1(t), \tau_2(t), . . . ,\tau_n(t)]^T$

$\underline{q}(t)$ = an nx1 vector of the joint position variables of the robot arm

$= [q_1(t) , . . . ,q_n(t)]^T$

$\dot{\underline{q}}(t)$ = an nx1 vector of the joint velocity of the robot arm

$= [q_1(t), . . . ,q_n(t)]^T$

$\ddot{\underline{q}}(t)$ = an nx1 acceleration vector of the robot joints

$= [q_1(t), . . . ,q_n(t)]^T$

$D(q)$ = an nxn symmetric inertia matrix

$\underline{h}(\underline{q}, \dot{\underline{q}})$ = an nx1 nonlinear coriolis and centrifugal force vector

$= [h_1, h_2, . . . ,h_n]^T$

$\underline{G}(q)$ = an nx1 gravity loading force vector

$= [g_1, g_2, . . . ,g_n]^T$

The inverse dynamics problem (IDA) can be solved using eqn. (4.1), that is, given the instantaneous joint position, velocity and acceleration, obtaining the required actuator torque. The expansion for the elements of D, $\underline{h}$, and $\underline{G}$ are given in Appendix A

Although computational effort required in calculating the dynamic coefficients in L.E. formulation is quite large, it is still preferable in off-line optimal path planning scheme because of its differential equation form which is very convenient for obtaining a state space model of the robot arm motion. State space model can be obtained by defining the state variables and inputs as follows ·

$$\begin{array}{lll} x_1 = \dot{q}_1 & x_{n+1} = q_1 & u_1 = \tau_1 \\ x_2 = \dot{q}_2 & x_{n+2} = q_2 & u_2 = \tau_2 \\ \vdots & \vdots & \vdots \\ x_n = \dot{q}_n & x_{2n} = q_n & u_n = \tau_n \end{array} \qquad (4\ 2)$$

With the above definition it follows that

$$\begin{aligned} \dot{x}_{n+1} &= x_1 \\ \dot{x}_{n+2} &= x_2 \\ &\vdots \\ \dot{x}_{2n} &= x_n \end{aligned} \qquad (4.3)$$

The vectors in eqn.(4.1) are now given by

$\underline{q} = [x_{n+1}, \ . \ . \ . \ . \ , x_{2n}]^T =$ position vector $=$ p

$\dot{\underline{q}} = [x_1, \ . \ . \ . \ . \ , x_n]^T =$ velocity vector $=$ v

$\underline{\tau} = [u_1, \ . \ \ . \ , u_n]^T =$ input vector $= \underline{u}$

Eqn. (4.1) can now be written as

$$u_i = \sum_{k=1}^{n} D_{i,k}\, \dot{x}_k + \sum_{j=1}^{n} \sum_{k=1}^{n} H_{i,j,k}\, x_j\, x_k + G_i \qquad (4.4)$$

$$i = 1, \ldots, n$$

With the above formulation , joint acceleration vector is given as

$$\ddot{\underline{q}} = [\ddot{q}_1, \ldots, \ddot{q}_n]^T = [\dot{x}_1, \ldots, \dot{x}_n]^T \qquad (4.5)$$

Here, it is not necessary to write the state eqn.(4.4) in the conventional form $\dot{\underline{x}} = \underline{f}(\underline{x},\underline{u},t)$ because in using MLV, given the system state trajectory between two points, we will need to find the inputs which will cause this state transition, and the form of eqn.(4.4) enables us to determine this directly.

4.2 METP Problem And Its Solution

4.2.1 Problem Formulation

Having the state space model of the robot manipulator optimal control problem is formulated as a fixed time, two point boundary value problem. The states (joint positions and velocities) are assumed to be specified at initial (t = 0) and final (t = T) points. Thus,

$$x_i(0) = x_i(T) = 0 \qquad i = 1, 2, \ldots, n \qquad (4.6a)$$

$$\left.\begin{aligned} x_i(0) &= x_{i0} \\ x_i(T) &= x_{iT} \end{aligned}\right\}, \quad i = n+1, \ldots, 2n \qquad (4.6b)$$

Eqn. (4.6a) stipulates that the initial and final velocities of the robot joints be zero. Eqn. (4.6b) stipulates the initial and final robot joint positions. Constraints on the magnitude of states and their derivatives are also specified at all $t \in [0,T]$. Input constraints, that is, constraints on magnitude of actuator torques are also given. These bounds are also specified as constant bounds of the form .

$$X_{i\ min} \le x_i \le X_{i\ max} \quad , \ i = n+1, \ . \ . \ . \ ,2n \qquad (4\ 7a)$$

This specifies bounds on joint positions .

$$X_{i\ min} \le x_i \le X_{i\ max} \quad , \ i = 1,\ 2, \ . \ . \ . \ ,n \qquad (4\ 7b)$$

This specifies bounds on robot joint velocities .

$$\dot{X}_{i\ min} \le \dot{x}_i \le \dot{X}_{i\ max} \quad , \ i = 1,\ 2, \ . \ . \ . \ ,n \qquad (4.7c)$$

This specifies bounds on robot joint accelerations .

$$\ddot{X}_{i\ min} \le \ddot{x}_i \le \ddot{X}_{i\ max} \quad , \ i = 1,\ 2, \ . \ . \ . \ ,n \qquad (4.7d)$$

This specifies bounds on robot joint jerks .

$$U_{i\ min} \le u_i \le U_{i\ max} \quad , \ i = 1,\ 2, \ . \ . \ ,n \qquad (4.7e)$$

This specifies bounds on motor torques .

The performance index for the problem is defined as follows :

$$J = \int_0^T \left[\sum_{i=1}^{n} \tau_i^2(t) \right] dt \qquad (4.8)$$

where

T = total time of traversal

$\tau_i(t)$ = Torque for the ith joint of the manipulator

Selection of this performance index is motivated by the fact that the servomechanism of each robot joint generally uses permanent magnet dc servo motors which are armature controlled So, the torque developed by each joint is proportional to the armature current as flux is constant. So, τ_i^2 is proportional to the copper loss in the ith joint motor and $\int_0^T \tau_i^2 \, dt$ is proportional to the energy lost in the ith joint motor as copper loss in the interval [O, T]. So, minimization of this performance index results in the minimization of electrical energy lost in the motors, assuming that the other losses in the motors are constant .

Now we will state the METP problem as follows :

Problem Statement- Find the optimal control $\underline{u}^*$ and optimal state trajectory $\underline{x}^*$ such that the performance index given by eqn (4.8) is minimized together with satisfaction of the end constraints given by eqn. (4.6) and constraints on states and inputs given by eqns. (4.7).

4.2.2 Solution To METP Problem

MLV algorithm presented in chapter 3 is used to solve the METP problem stated above.

First of all an initial feasible cartesian path is chosen from the initial point to the final point. In order to facilitate choice of a feasible trajectory, the time of traversal is chosen longer than dictated by minimum traversal time consideration so that the effects of coriolis, centrifugal forces etc are small and all the constraints are satisfied without much difficulty. Some points are selected on this cartesian path and corresponding joint configurations are obtained using IKA. Then B-Splines are used to fit these points and the corresponding joint trajectories are obtained. Trajectories for joint velocities, accelerations etc are also obtained using the spline coefficients for the derivatives. Now, all the steps of the MLV algorithm are followed starting with a suitable value of Delta In Step 10 constraints are taken as those specified in eqns (4 6) and (4.7). The algorithm is applied for more than one starting feasible trajectory and different perturbation values are also used.

4.3 Results Of Computer Simulation

The MLV algorithm for the solution of METP problem was verified through the computer simulation of first three joints of PUMA 560 robot. Two examples are considered.

4.31 METP Using Cubic B-Splines

Problem 1- Plan the optimal robot joint trajectories and corresponding time sequences of joint velocities and accelerations to move the EE from (0.38m, 0.218, -0.092) to (-0.405, -0.194, -0.04) in cartesian coordinates in a fixed time. Find the required actuator torques also as functions of time.

Solution - As only first three joints of PUMA 560 are considered, it is not needed to specify the orientation of the EE, which is at the third joint.

For the selection of the feasible trajectory a time of traversal of 15.5 secs was found to be sufficient 10 knots were selected in between and the EE configuration at these knot points was converted into corresponding joint configurations using IKA (Table 4.1). The initial and final velocities of all the robot

Table 4.1

Knot sequence and corresponding initial data sequence for Example 1.

Joint Knot(sec)	1st	2nd degrees	3rd
0	10	15	60
1	25	20	80
2	60	25	135
3	75	30	155
5	130	-45	75
7	100	-55	40
9	70	-70	-10
11	20	-60	20
13	-30	-40	65
14	-40	-15	85
15	-45	5	70
15.5	-45	8	69

Table 4.2

Constraints on joint position, velocity, acceleration, jerk and actuator torque

Joint / Constraints	1st	2nd	3rd
Position (degree)	±160	-225 to 45	-45 to 225
Velocity (deg/sec)	±100	±95	±100
Acceleation (deg/sec^2)	±50	±60	±70
Jerk (deg/sec^3)	±98	±98	±92
Torque (N-m)	±70	±100	±70

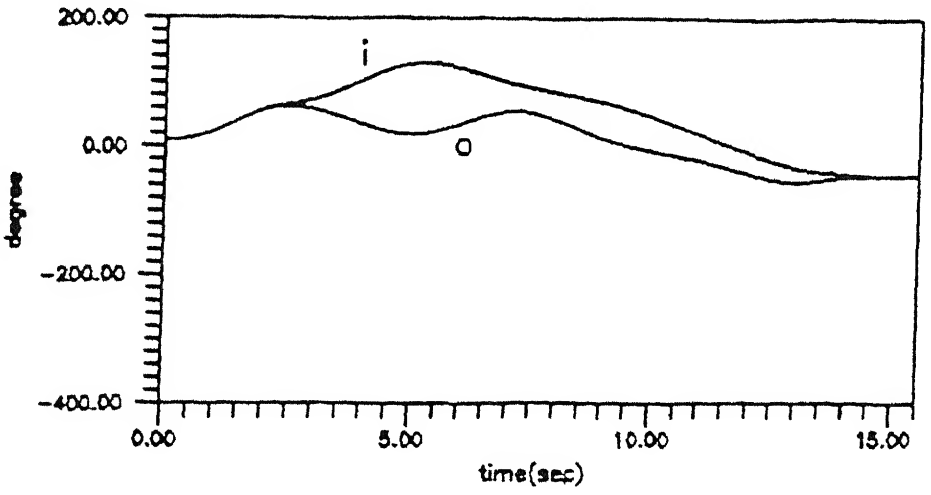
joints were assumed to be zero This is because it is assumed that the robot arm starts at rest and comes to rest at the end of the trajectory. Now, B-Splines were used to fit these knots. Selected knot set and extended knot set were the same as those selected in Example 1 in chapter 3. So, the same spline matrix was used because only the vector $\underline{f}$ in eqn (3.4) was changed. Matrix B remained unchanged as it depends only on the knot sets and data points. This is another advantage of B-Splines

The bounds on the joint position, velocity and acceleration are shown in Table 4.2. Torque bounds are also shown in Table 4.2. The time of traversal was divided into 200 equal intervals for the purpose of calculating the performance index using the trapezoidal rule. The initial value of the performance was found to be 36639 units.

Now a value of 30 was used for Delta and 1 for Del_min MLV was applied to perturb the nominal trajectory and after 27 iterations the value of performance index was found to be 14468 units. The initial and optimal trajectories for joint positions, velocities and accelerations are shown in Figs. (4.1) to (4.9). Corresponding optimal actuator torques are shown in Figs. (4 10) to (4 12). Variations in performance index are shown in Fig. (4.13) against iteration number.

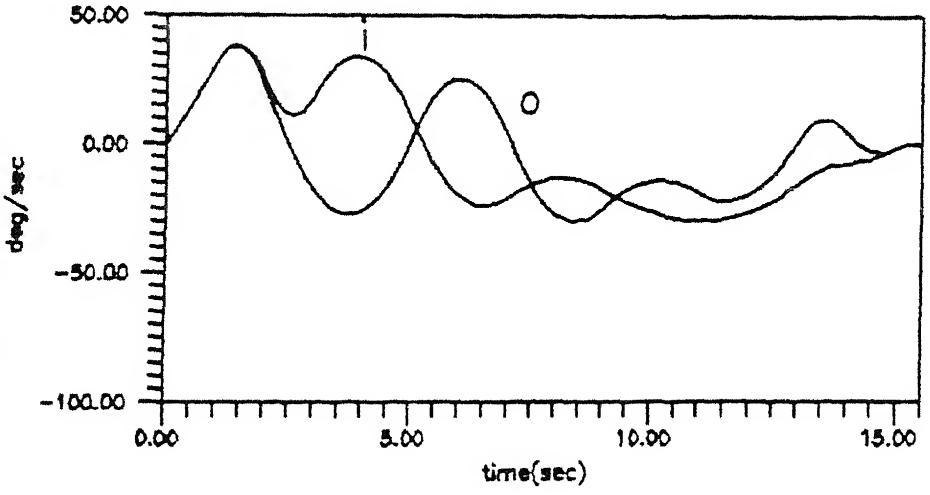
Again, another nominal trajectory was selected and perturbed, starting with a value of Delta as 15. The performance index was reduced from an initial value of 19361 units to the final value of 18022 units in 15 iterations

Fig 4.1 – POSITION PLOT OF JOINT 1



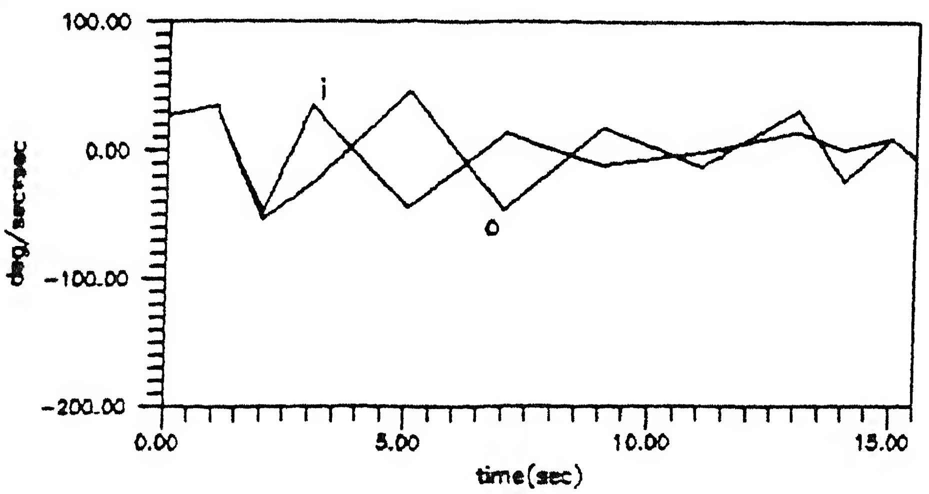
i – initial , o – optimal

Fig 4.2 – VELOCITY PLOT OF JOINT 1



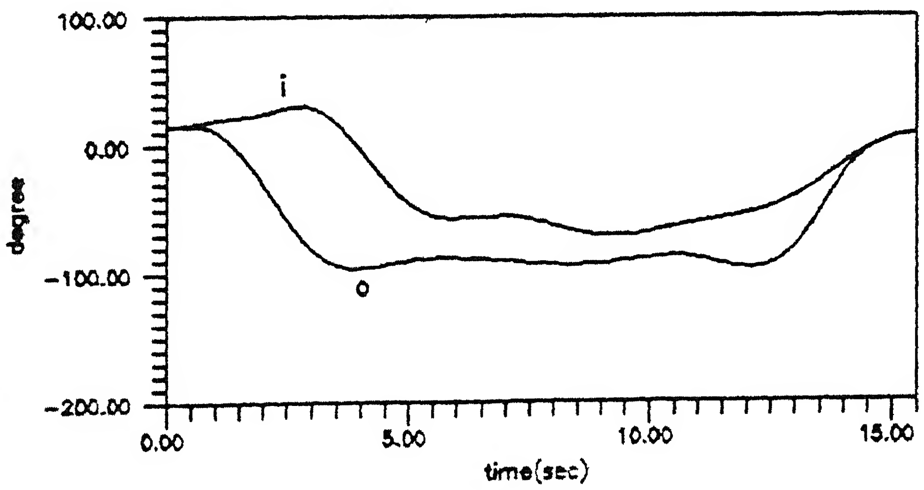
i – initial , o – optimal

Fig 4.3 – ACCELERATION PLOT OF JOINT 1



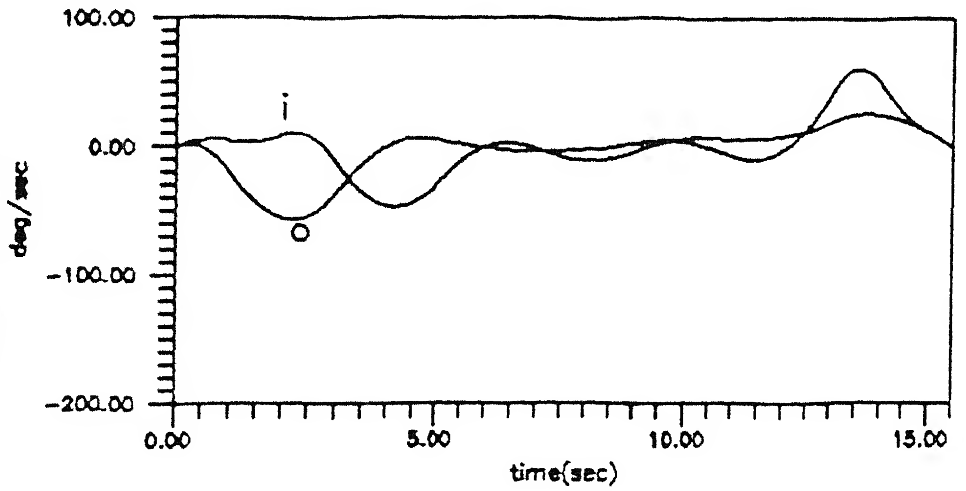
i – initial , o – optimal

Fig 4.4 – POSITION PLOT OF JOINT 2



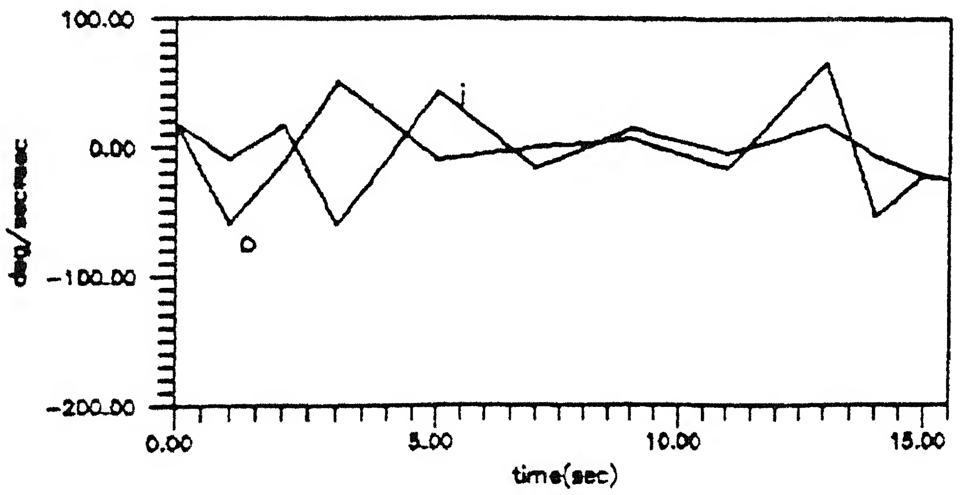
i – initial , o – optimal

Fig 4.5 – VELOCITY PLOT OF JOINT 2



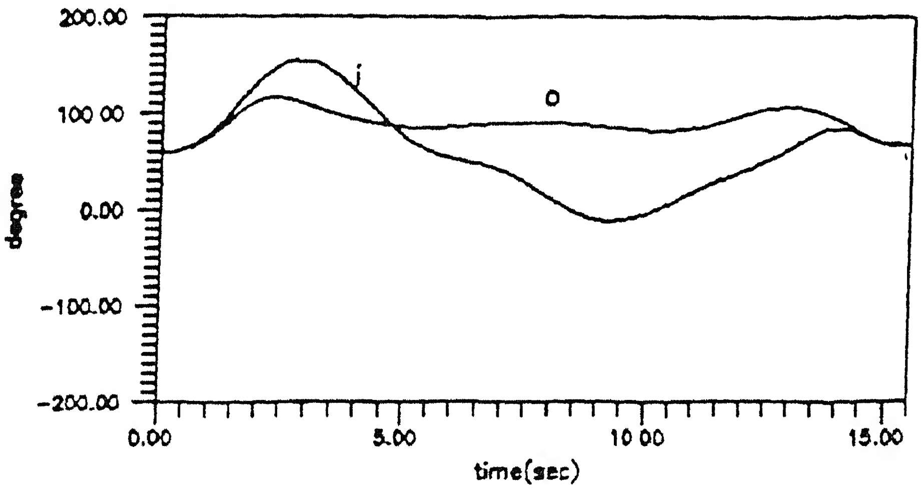
i – initial , o – optimal

Fig 4.6 – ACCELERATION PLOT OF JOINT 2



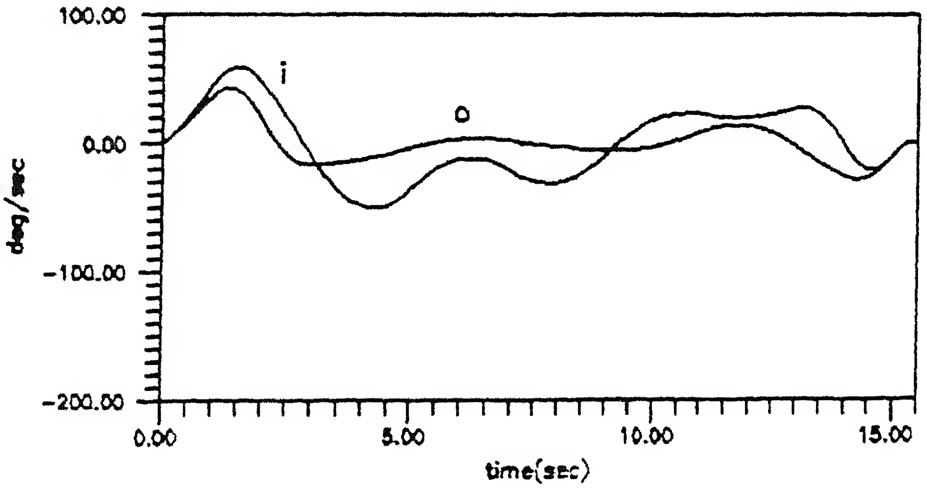
i – initial , o – optimal

Fig 4.7 – POSITION PLOT OF JOINT 3



i – initial , o – optimal

Fig 4.8 – VELOCITY PLOT OF JOINT 3



i – initial , o – optimal

Fig 4.9 – ACCELERATION PLOT OF JOINT 3

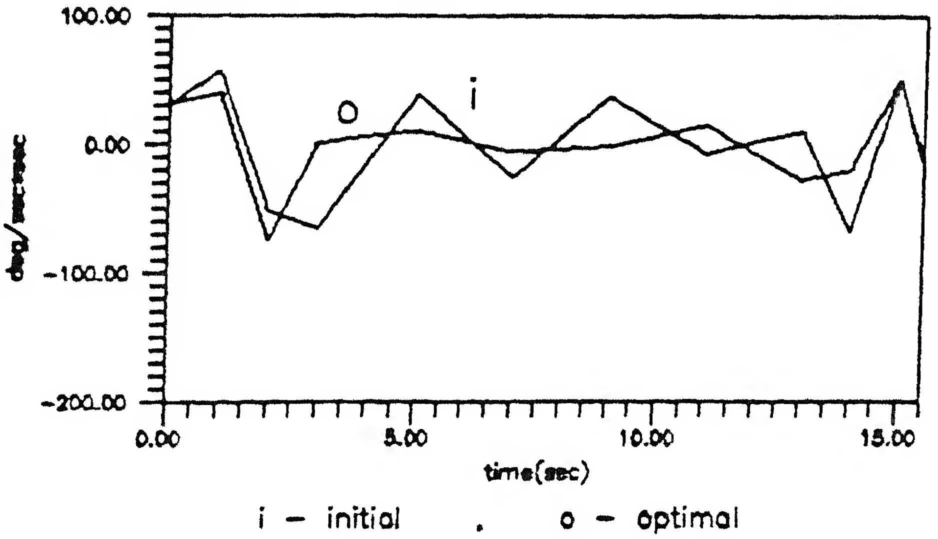


Fig 4.10 – TORQUE PLOT OF JOINT 1

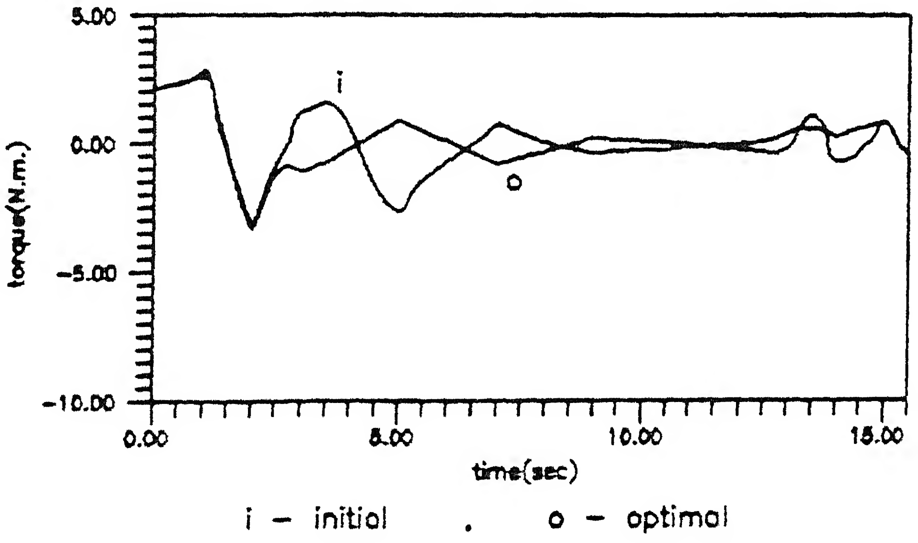
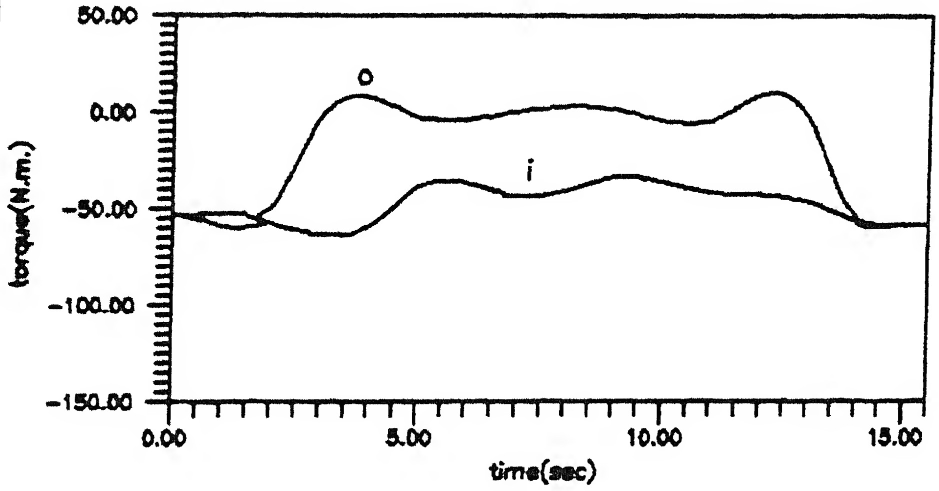
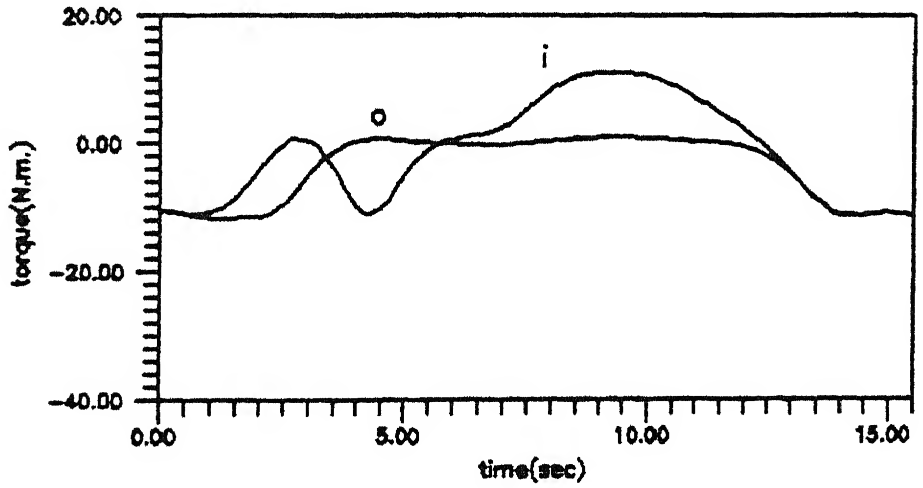


Fig 4.11 – TORQUE PLOT OF JOINT 2



i – initial , o – optimal

Fig 4.12 – TORQUE PLOT OF JOINT 3



i – initial , o – optimal

Fig 4.13 – CONVERGENCE CURVE FOR K = 4

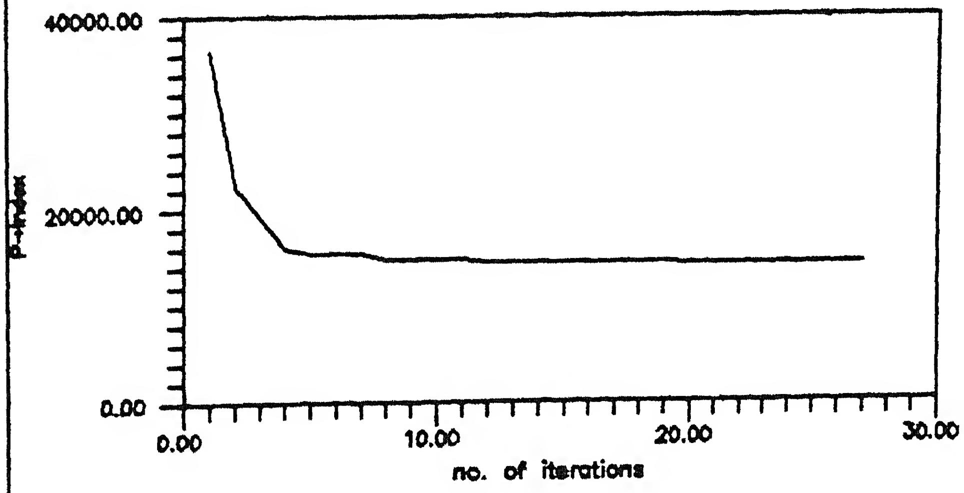


Fig. 4.14 CONVERGENCE CURVE FOR K = 5

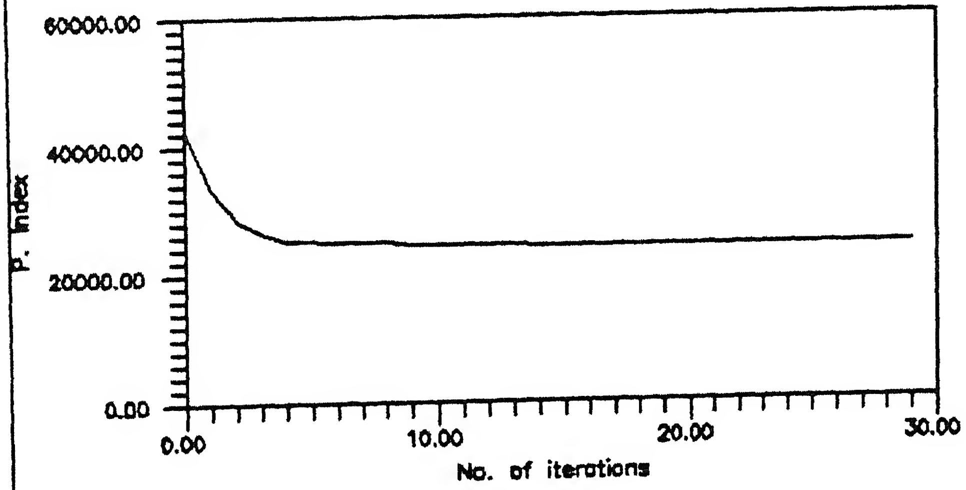
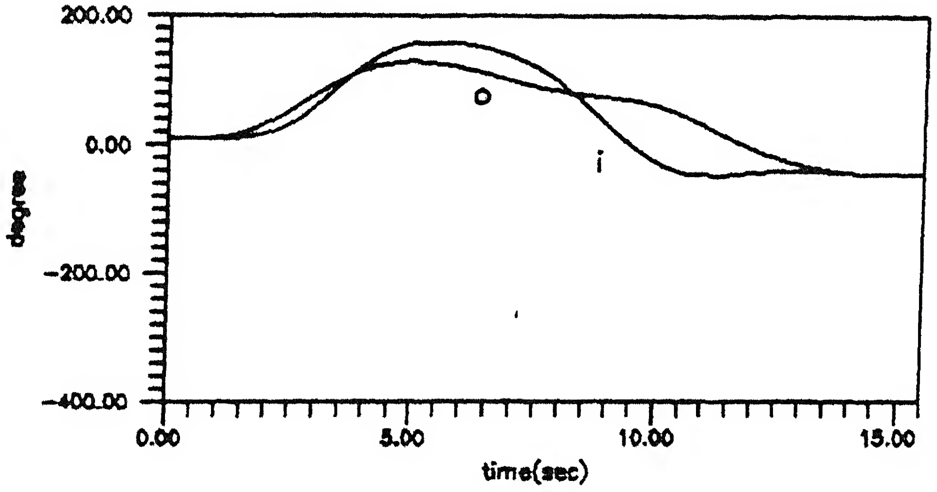


Table 4.3

Knot sequence and corresponding initial data sequence for Example 2.

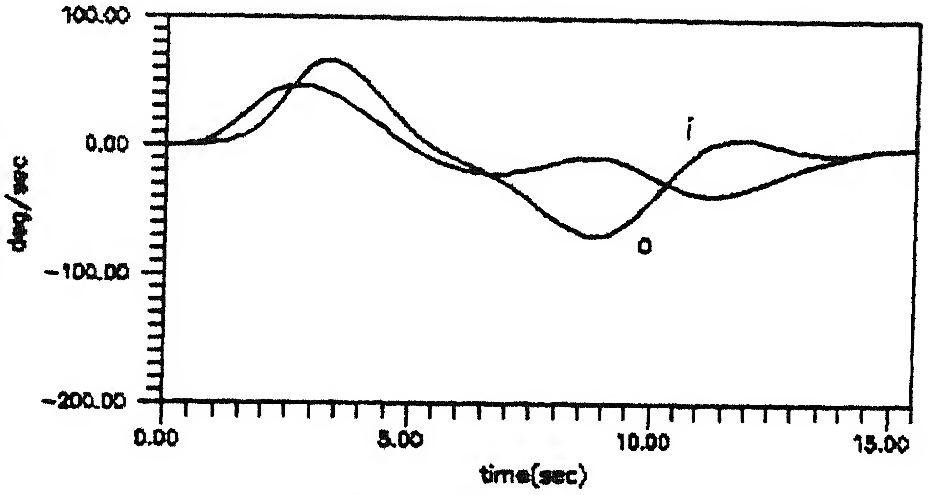
Joint / Knot(sec)	1st	2nd	3rd
	degrees		
0	10	15	60
1	12	17	63
2	35	27	90
3	80	39	135
5	130	-50	180
7	100	-136	165
9	75	-70	200
11	30	-5	160
13	-30	15	75
14	-42	9	70
15.5	-45	8	69

Fig 4.15 POSITION PLOT OF JOINT 1



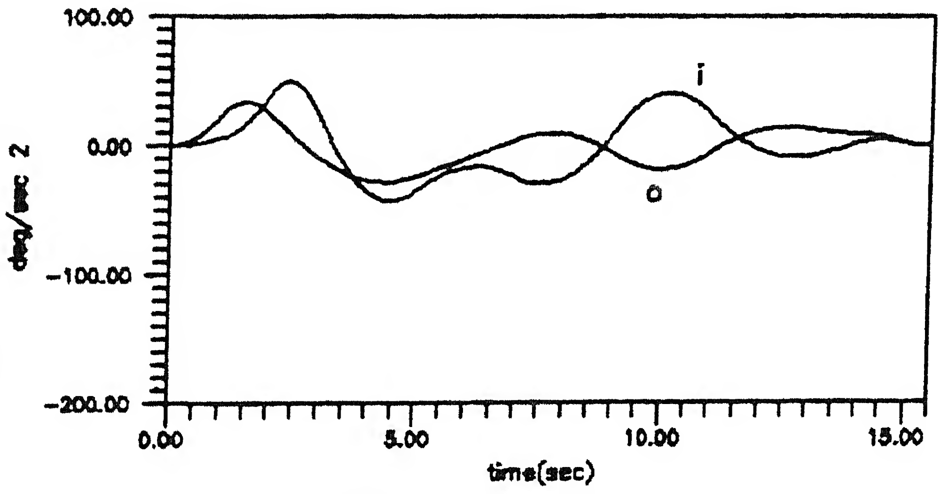
i – initial , o – optimal

Fig 4.16 VELOCITY PLOT OF JOINT 1



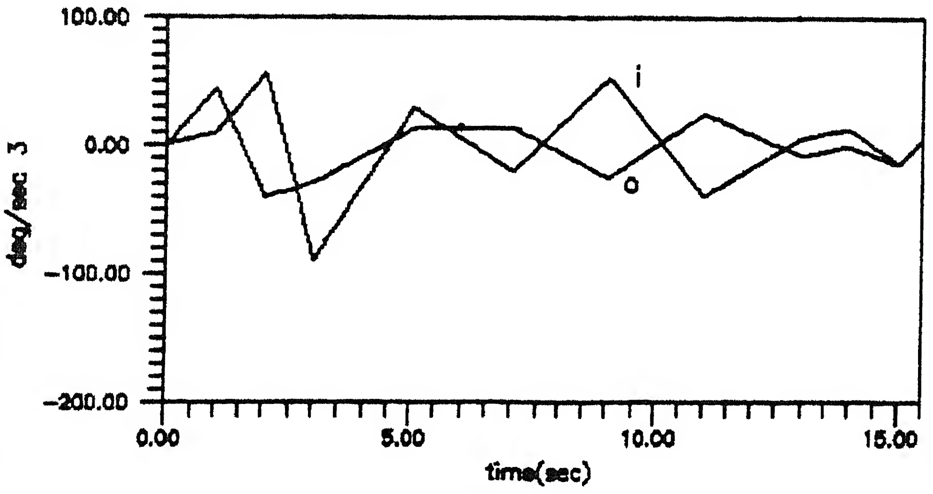
i – initial , o – optimal

Fig 4.17 ACCELERATION PLOT OF JOINT 1



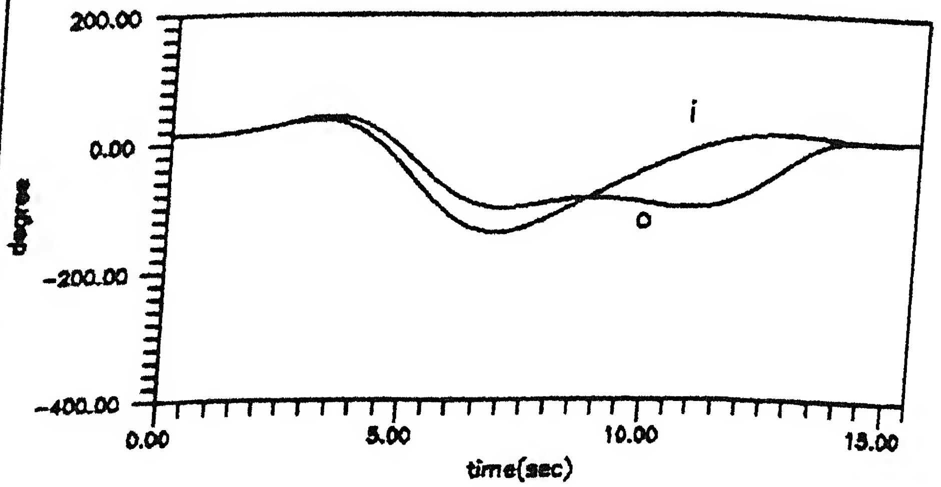
i – initial , o – optimal

Fig 4.18 JERK PLOT OF JOINT 1



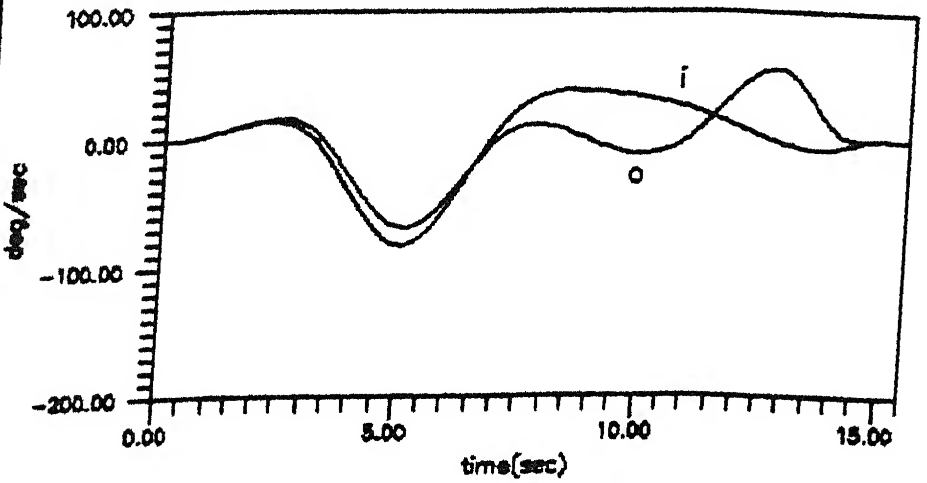
i – initial , o – optimal

Fig 4.19 POSITION PLOT OF JOINT 2



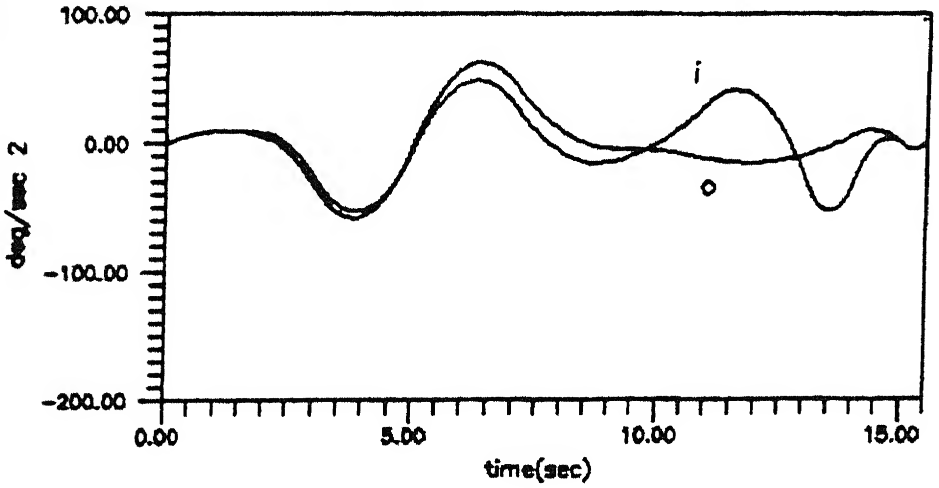
.i – initial , o – optimal

FIG 4.20 VELOCITY PLOT OF JOINT 2



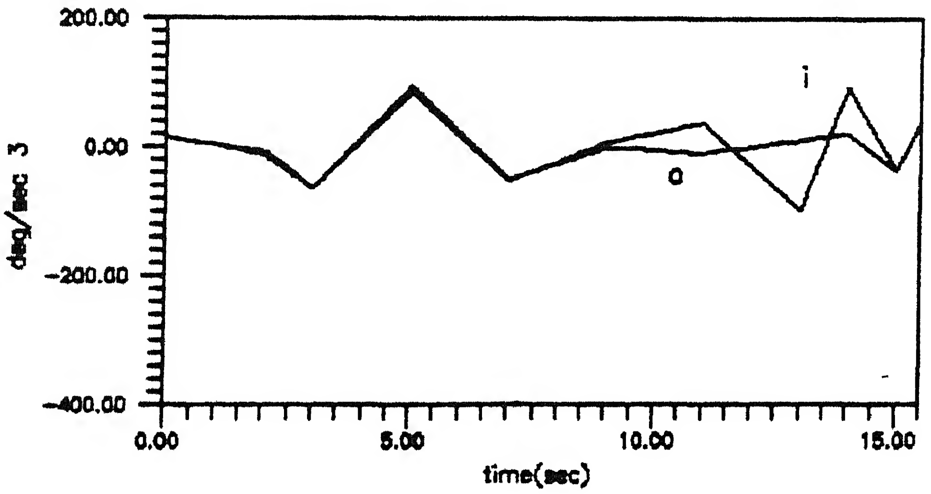
i – initial , o – optimal

Fig 4.21 ACCELERATION PLOT OF JOINT 2



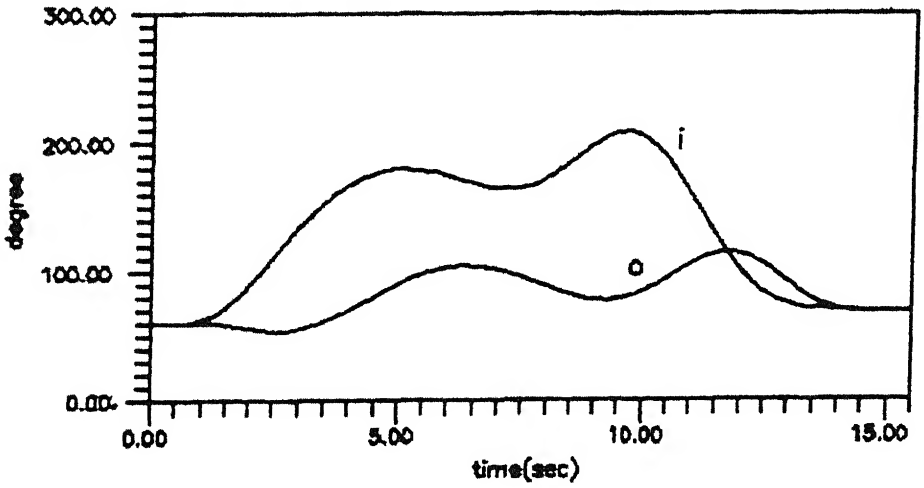
i – initial , o – optimal

Fig 4.22 JERK PLOT OF JOINT 2



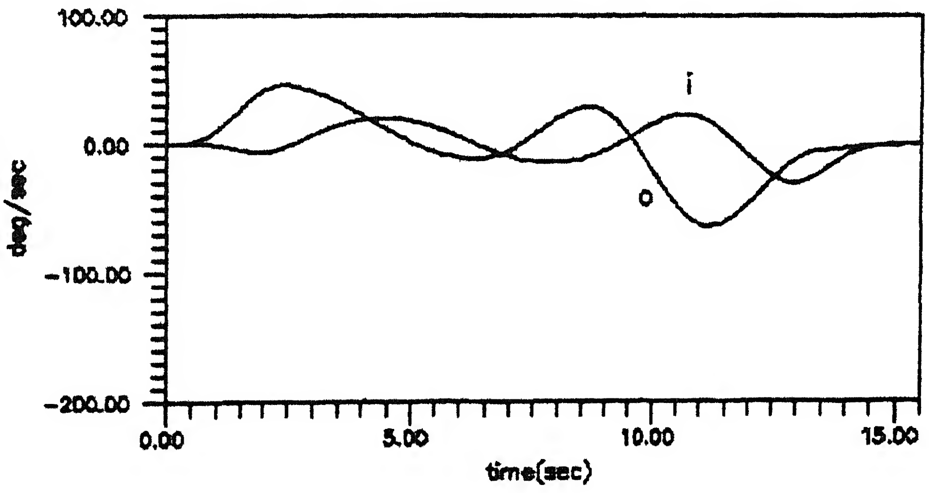
i – initial , o – optimal

Fig 4.23 POSITION PLOT OF JOINT 3



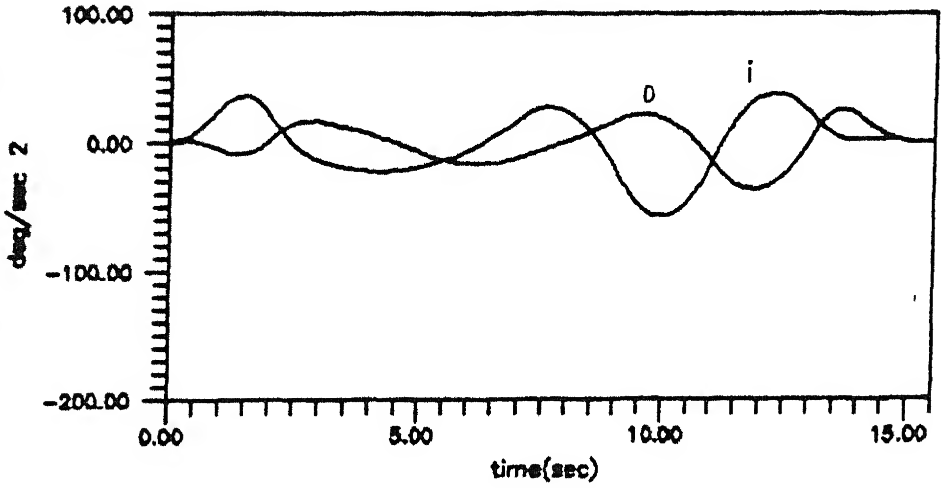
i – initial , o – optimal

Fig 4.24 VELOCITY PLOT OF JOINT 3



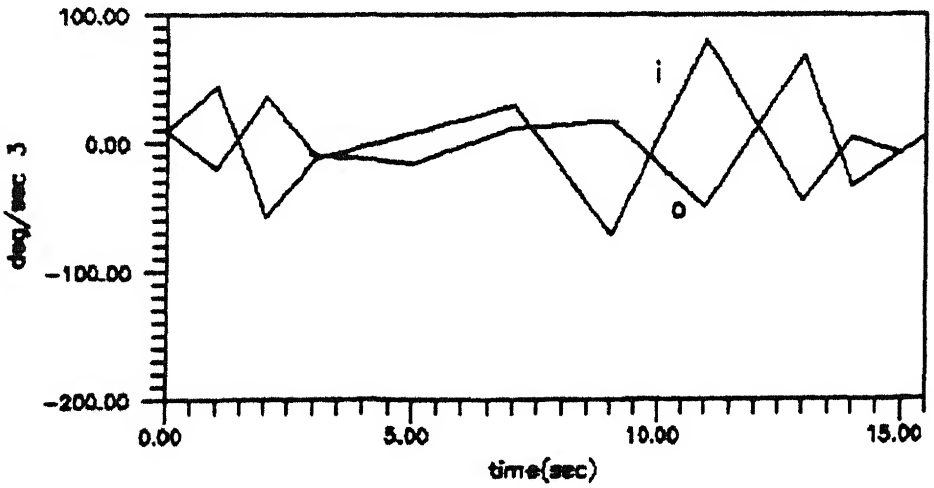
i – initial , o – optimal

Fig 4.25 ACCELERATION PLOT OF JOINT 3



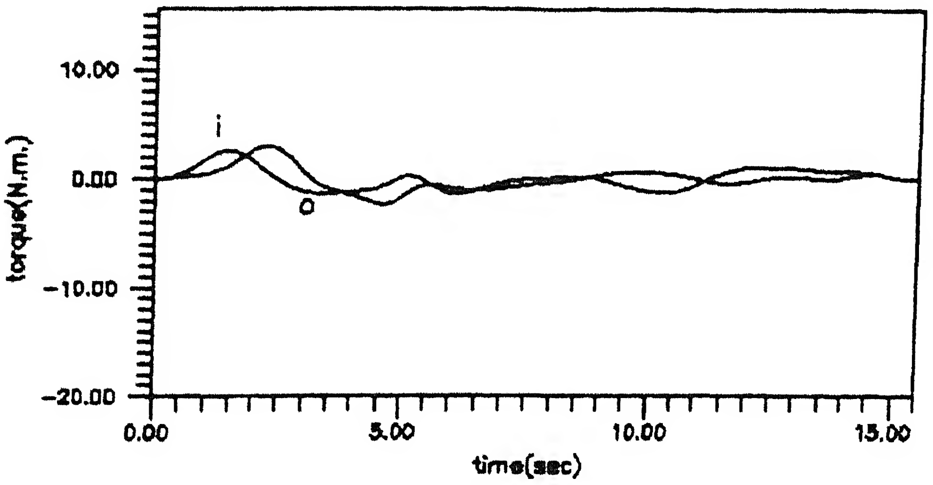
i – initial , o – optimal

Fig 4.26 JERK PLOT OF JOINT 3



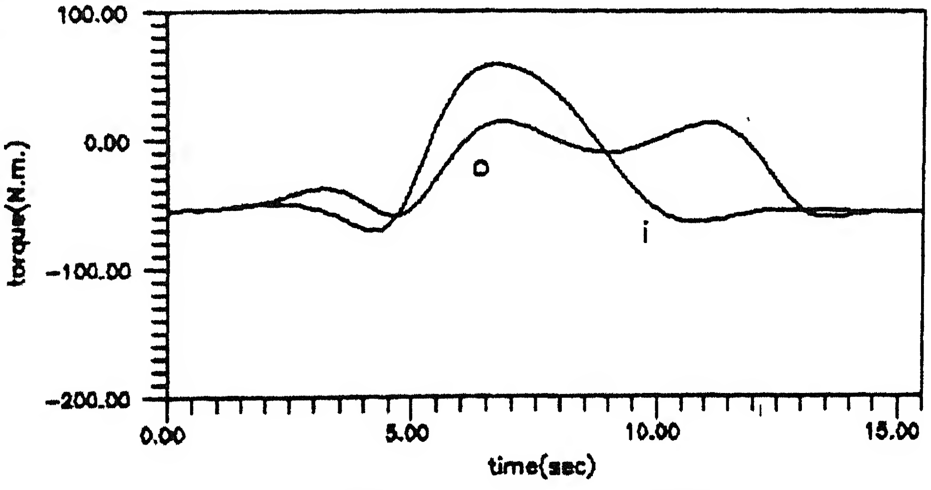
i – initial , o – optimal

Fig 4.27 TORQUE PLOT OF JOINT 1



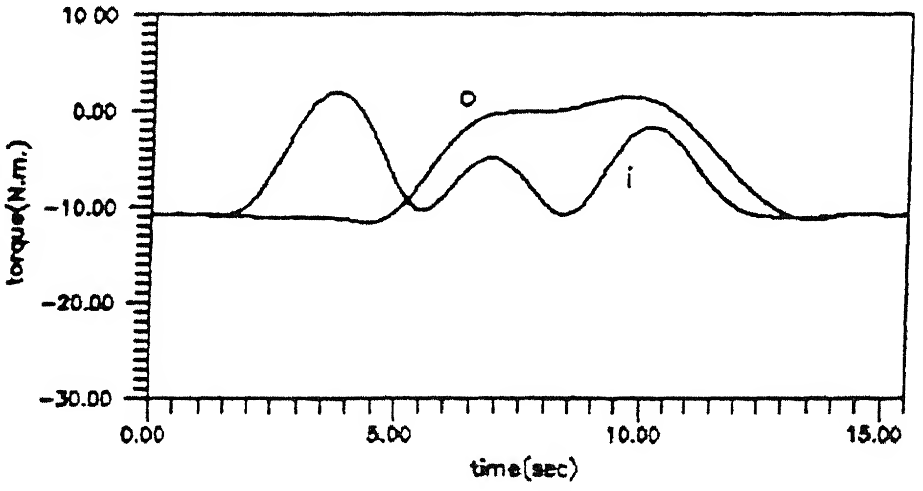
i – initial , o – optimal

Fig. 4.28 TORQUE PLOT OF JOINT 2



i – initial , o – optimal

Fig. 4.29 TORQUE PLOT OF JOINT 3



i – initial , o – optimal

4.4 CONCLUSIONS

From the results obtained above, it is evident that the METP algorithm finds a local minimum and it depends on the starting trajectory and perturbation values. In order to capture the absolute minimum we must try different initial trajectories and different perturbation values.

Also, it is clear that the performance index reduces very fast in first few iterations and after that its convergence is sluggish. So, we can stop after few iterations with a significant reduction in performance index.

CHAPTER 5

MINIMUM TIME TRAJECTORY PLANNING

In the present industrial environment, manipulators are required to be operated at high speed, that is, total time of manipulator movement needs to be minimized to perform a particular task. Minimum Time operation is specially important in the case where robot is required to perform some repetitive job. Conventional trajectory planning schemes generally deal with MTTP only. The usual approach is to first generate a non-optimal geometric path and then minimize the traversal time of the EE movement along this path. Thus, traditionally, geometric path is not changed during optimization .

In this chapter the solution to MTTP problem has been obtained by using MLV. An algorithm has been developed and applied to spline interpolated trajectory to find a time optimal path from an origin to a destination . This algorithm is based on successively shrinking the whole knot sequence of the interpolation interval and applying MLV to the spline coefficients to maintain the feasibility of the joint trajectories.

5.1 Problem Formulation

The problem formulation assumes the state space model given by eqn. (4.4) for the dynamics of the robot arm motion . This is reproduced here for convenience .

$$u_i = \sum_{k=1}^{n} D_{i,k}\,\dot{x}_k + \sum_{j=1}^{n}\sum_{k=1}^{n} H_{i,j,k}\,x_j\,x_k + G_i \qquad (5.1)$$

$$i = 1, \ldots, n .$$

Joint positions and velocities are assumed to be specified at the initial (t = 0) and final (t = T) time instants , that is ,

$$x_i(0) = x_i(T) = 0 , \quad i = 1, \ \ldots ,n \qquad (5.2a)$$

$$\left.\begin{aligned} x_i(0) &= x_{i0} \\ x_i(T) &= x_{iT} \end{aligned}\right\} , \quad i = n+1 , \ldots ,2n \qquad (5.2b)$$

where x_{i0} , x_{iT} , $i = n+1, \ldots ,2n$ are specified values Also the constraints on state variables (position, velocity, acceleration and jerk) and those on control inputs are specified as constant bounds :

$$X_{i\ min} \le x_i \le X_{i\ max} , \quad i = n+1, \ldots ,2n \qquad (5.3a)$$

$$X_{i\ min} \le x_i \le X_{i\ max} , \quad i = 1, 2 ,\ldots ,n \qquad (5.3b)$$

$$\dot{X}_{i\ min} \le x_i \le \dot{X}_{i\ max} , \quad i = 1, 2, \ldots ,n \qquad (5.3c)$$

$$\ddot{X}_{i\ min} \le \ddot{x}_i \le \dot{X}_{i\ max} , \quad i = 1, 2, \ldots ,n \qquad (5.3d)$$

$$U_{i\ min} \le u_i \le U_{i\ max} \qquad (5.3e)$$

It is desired to find the optimal path, (time sequence of position, velocity, acceleration etc.), and desired control inputs so that the final time T is minimized. The MTTP problem can now be stated as follows :

Problem Statement - Find the optimal state $\underline{x}^*(t)$ and the corresponding optimal control input $\underline{u}^*(t)$ satisfying eqn.(5.1) so that the time of traversal is minimized, with the end point constraints (eqn.(5.2)) and state and input constraints (eqn. (5.3)) satisfied.

5.2 MTTP Algorithm

The following notations are used :

K = Order of the B-Spline

$a_{j,i,k1}$ = k1th B-Spline coefficient for jth derivative of the ith state variable . j = 0, 1, . . . ,K-2 ; i = 1, . . . ,n k1 = 1, . . . ,m+K-1 . $a_{0,i,k1}$ represents coefficients for the ith state trajectory .

P = Saturation indicator for a particular value of the variable labelled Delta . It indicates the number of successful local variations for a particular Delta . P = 0 indicates that there have not been any successful local variations and implies that the current value of delta should be changed .

r = Saturation indicator for the whole range of Delta. r = 0 indicates that none of the variations is valid in the current iteration and saturation is reached, that is, final time T has reached the minimum value and no further reduction in it is possible.

Delta = amount of perturbation in a particular B-Spline coefficient in the state trajectory representation.

Del-min = minimum permissible perturbation in a B-Spline coefficient in the state trajectory representation.

$\zeta(k)$ = A particular knot point of the extended knot set, k = 1,2, . . . ,m+2K-1

$[t_b,t_e]$ = Interval affected by a particular B-Spline coefficient

The MTTP algorithm using B-Spline representation is as follows :

Step 1 : Obtain a B-Spline approximation of feasible state trajectories and their (K-2) derivatives with respect to the extended knot set ($\zeta[k]$; $k = 1, \dots, m+2K-1$)

where

$$m+1 = \text{total number of original knot points}$$

Store the spline coefficients $a_{j,i,k1}$. $j = 0, \dots, K-2$; $i = 1, \dots, n$; $k1 = 1, \dots, m+K-1$.

Step 2 : Set $r = 0$

Step 3 : For $k = K + 1$ to $m + 2K - 1$ do

$$\zeta[k] = \zeta[k] - \Delta Z \cdot f$$

where

$$\Delta Z = \zeta[k] - \zeta[k-1]$$

$$f = \text{constant} < 1$$

Set $T = \zeta[m+K]$

Step 4 : Set $l = 1$, Delta, C

Step 5 : Set $P = 0$

Step 6 : Set $i = 1$

Step 7 : Set $k1 = c_{IN}$, where a_1, a_2, . . . , $a_{c_{IN}-1}$ are the coefficients contributing to the function representation at the initial point.

Step 8 : Determine the interval $[t_b, t_e]$ affected by $a_{0,i,k1}$.

Step 9 : Set $a = 1$

Step 10: Delta = a*Delta

$a_{0,i,k1} = a_{0,i,k1}$ + Delta

Step 11: Calculate the ith state , its derivatives and associated controls in the interval $[t_b, t_e]$ with the new values of

```
          $a_{j,i,k1}$, [j = 0, . . . ,K-2] and check whether
          (i)the new states are admissible and
          (ii)the control inputs required to translate the states
          in the interval [$t_b$,$t_e$] along the changed trajectory are
          admissible
                    If 'Yes' then go to Step 13 else go to Step 12.
Step 12:  Restore the coefficients
                    $a_{0,i,k1} = a_{0,i,k1} - Delta$
                    Set a = -a
          If a = -1 then go to Step 10 else go to Step 14.
Step 13:  Success met
                    P = P + 1
Step 14:  k1 = k1 + 1
          If k1 ≤ $C_{FN}$, then go to Step 8,  where $a_{C_{FN+1}}$, . . .
          ,$a_{m+K-1}$ are the coefficients contributing to the function
          representation at the final point
Step 15:  i = i + 1 ;   if i ≤ n then go to Step 7.
Step 16:  l = l + 1  ;  r = r + P ;
          if P ≢ 0 then go to Step 5.
Step 17:  Set Delta = Delta/C
          If Delta > Del_min then go to Step 2.
Step 18:  If r ≢ 0 then go to Step 2.
Step 19:  Stop.
```

5.3 Solution To MTTP Problem

Some important steps of the above algorithm must be clear for the solution of MTTP problem.

Here for the interpolation of the state trajectories using B-Splines dummy knots are selected as follows:

$$\zeta_1 = \zeta_2 = \ . \ . \ . = \zeta_{K-1} = 0$$

$$\zeta_{m+K-1} = \ . \ . \ . = \zeta_{m+2K-1} = T$$

This is done purposely because, otherwise, during the iteration process, shrinking of the knot sequence may result in violation of the state constraints at the initial and final points. By making the dummy knots same as the end knots, shrinking of the knots does not violate the end point constraints.

After selecting a suitable interpolation interval and a (preferably) feasible trajectory on it, joint trajectories are subjected to MLV algorithm Knots are shifted by equal percentage of their separation and then spline coefficients are varied to check all the constraints.

5.4 Computer Simulation Results

The above algorithm has been verified by computer simulation of first three joints of PUMA 560 robot manipulator. A simulation example is given below.

Example 1 - An initial time of traversal of 27.5 seconds was chosen to move the EE from a point (0.253m, 0.359m, -0.129m) to the final point (-0.048m, 0.456m, 0.0 6m) in cartesian space. The original knot set and extended knot set were selected as follows -

Original Knot Set - 0, 2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20, 22.5, 25, 27.5.

Table 5.1

Knot sequence and corresponding initial

data sequence for Example 1

Joint Knot(sec)	1st	2nd (degrees)	3rd
0	35	20	90
2.5	45	25	80
5.0	70	15	100
7.5	90	-5	150
10.0	100	30	180
12.5	80	-80	190
17.5	75	-110	170
20.0	10	-140	140
22.5	-30	-150	130
25.0	-70	-160	100
27.5	-65	-170	95

Fig. 5.1 POSITION PLOT OF JOINT 1

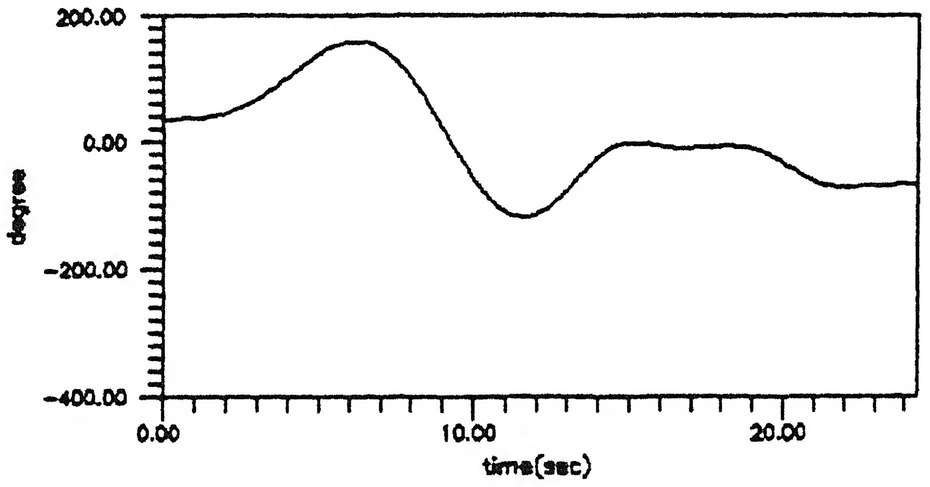


Fig. 5.2 VELOCITY PLOT OF JOINT 1

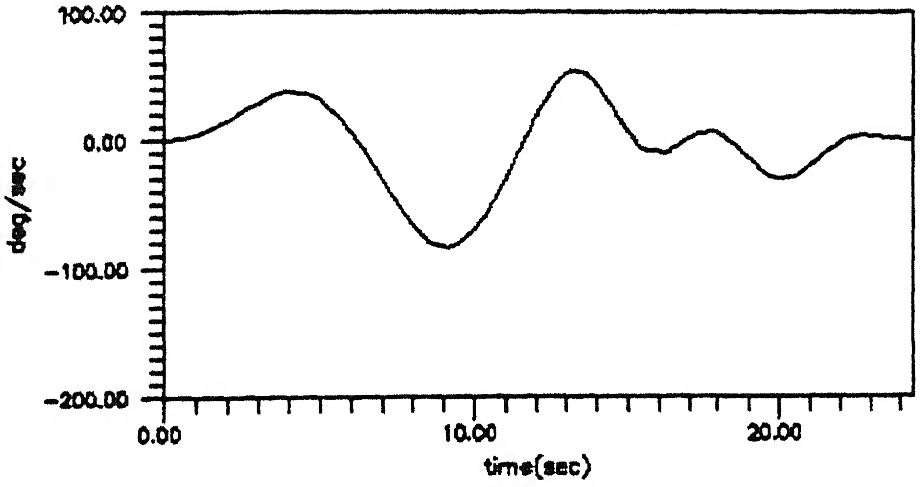


Fig. 5.3 POSITION PLOT OF JOINT 2

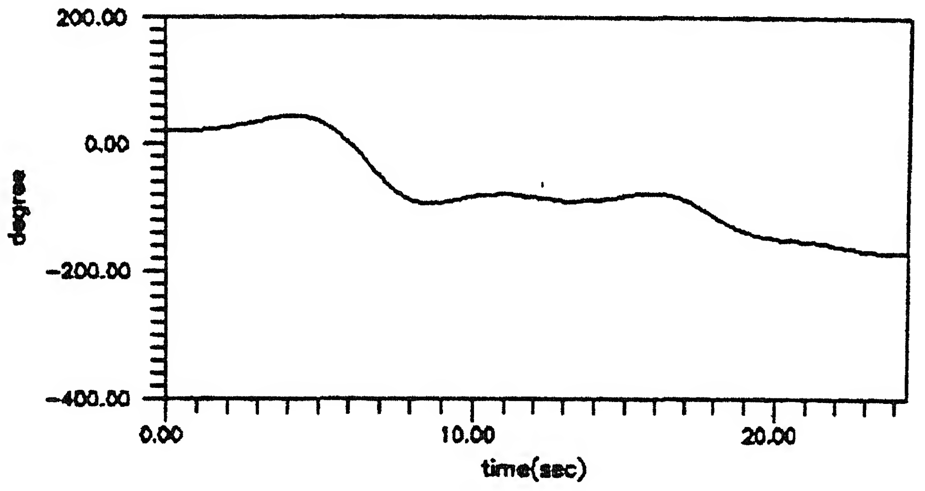


Fig. 5.4 VELOCITY PLOT OF JOINT 2

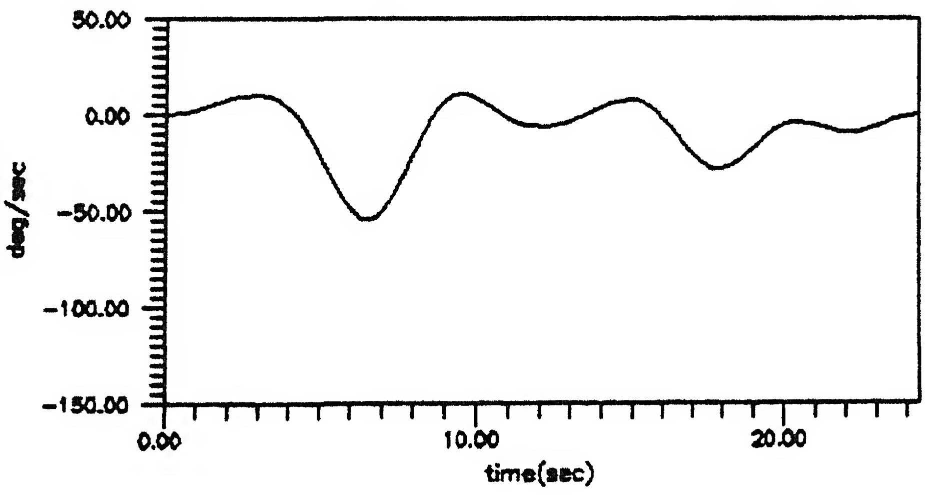


Fig. 5.5 POSITION PLOT OF JOINT 3

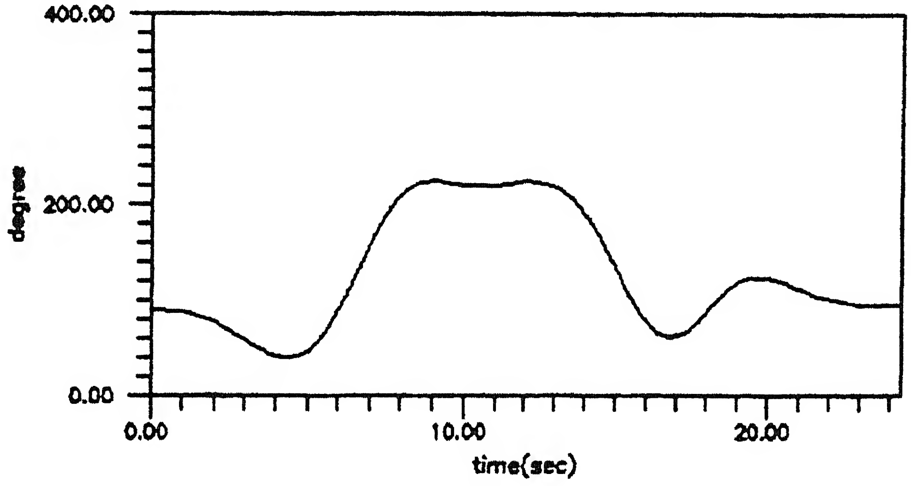


Fig. 5.6 VELOCITY PLOT OF JOINT 3

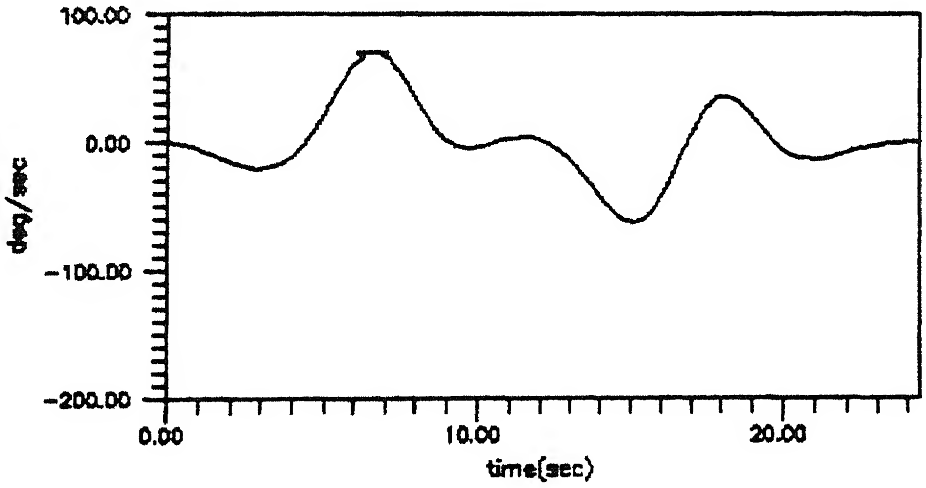


Fig. 5.7 TORQUE PLOT OF JOINT 1

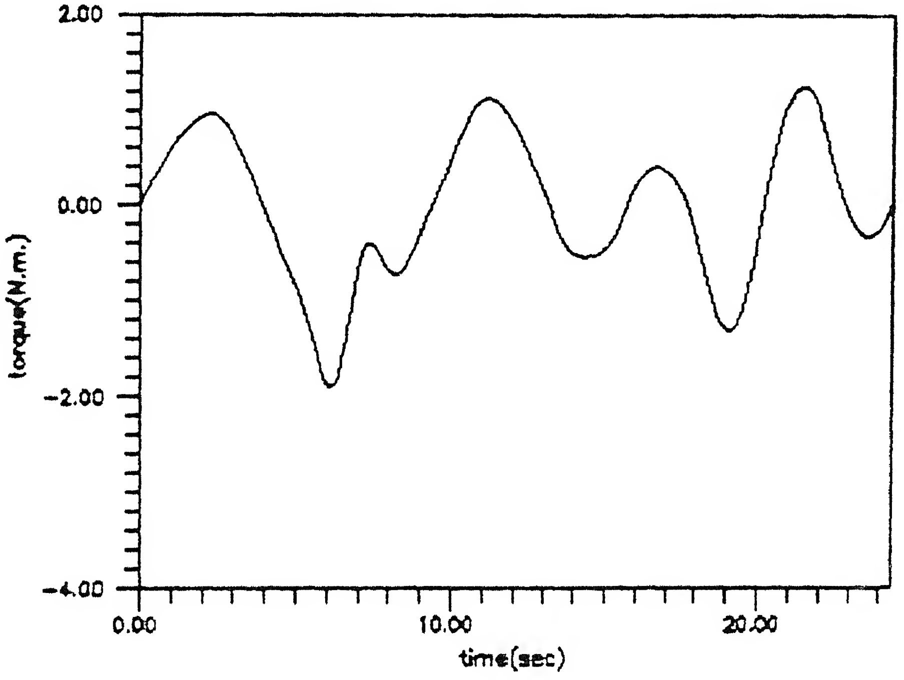


Fig. 5.8 TORQUE PLOT OF JOINT 2

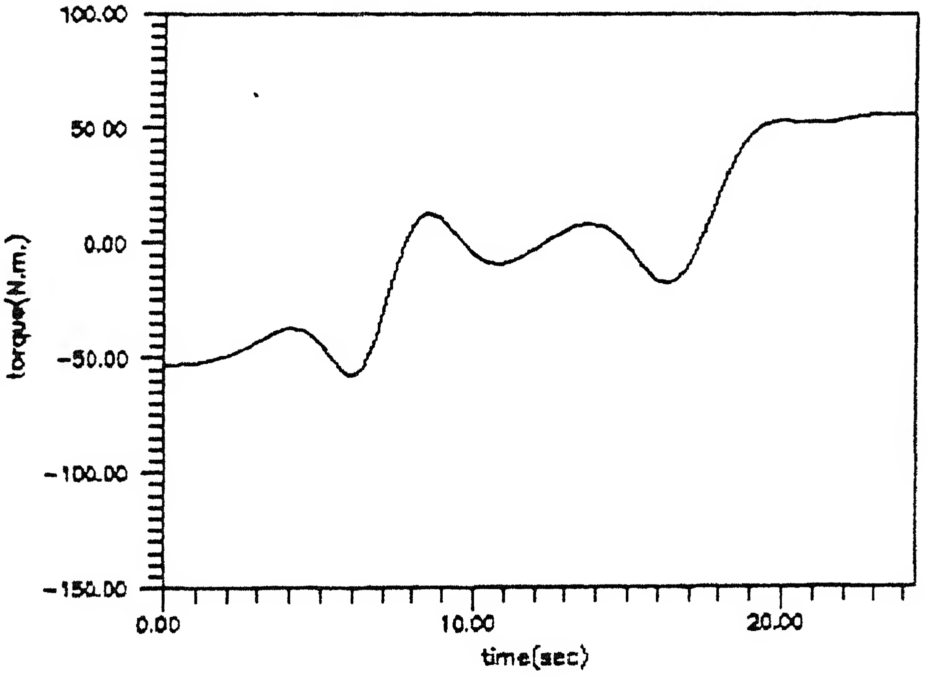
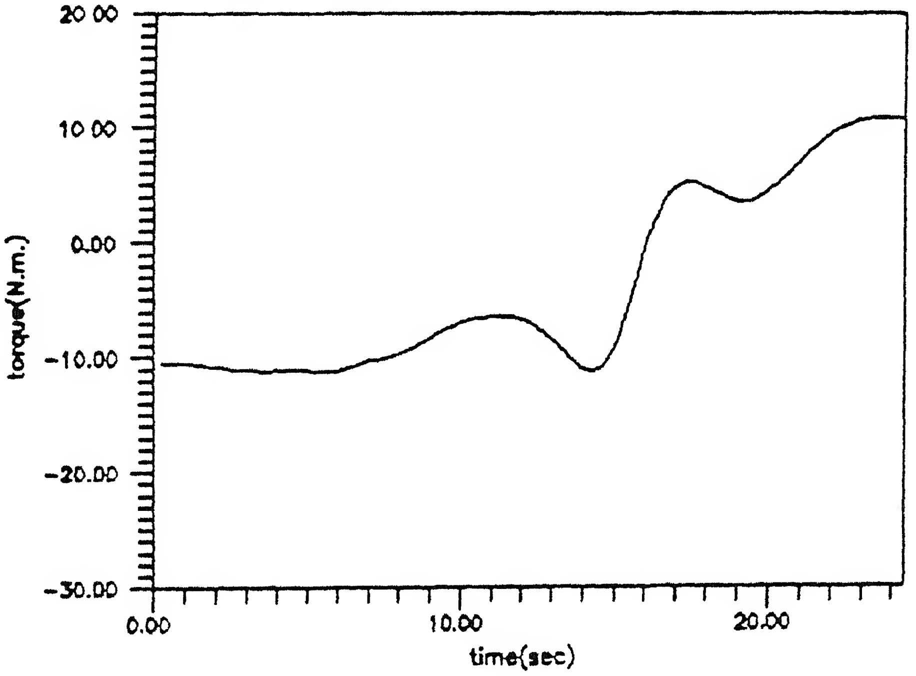


Fig. 5.9 TORQUE PLOT OF JOINT 3



Extended Knot Set - 0, 0, 0, 0, 0, 2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20, 22.5, 25, 27.5, 27.5, 27.5, 27.5, 27.5.

Data Point Set - 0, 0, 0, 2.5, 5, 7.5, 10, 12.5, 17.5, 20, 22.5, 25, 27.5, 27.5, 27.5.

Initial and final velocity constraints have been included in 2nd and 14th equations respectively. Initial and final acceleration constraints have been included in 3rd and 13th equations respectively. Initial joint values, at the data points, obtained from inverse kinematic analysis are shown in Table 5.1. Continuous joint trajectories are obtained by fitting these data points and MTTP algorithm has been applied. Delta was chosen as 30 and reduced by a factor of 1.5 at each step. Del_min was selected as 1. Minimum time obtained was 24.376 seconds. The final extended knot sequence was

$\{0, 0, 0, 0, 0, 2.216, 4.432, 6.648, 8.864, 11.08, 13.296, 15\ 512, 17.728, 19.944, 22.16, 24.376, 24.376, 24.376, 24.376, 24.376\}$

The optimal trajectories for joint positions and velocities are shown in Figs.5.1 to 5.6. Torque plots are also given in Figs. 5.7 to 5.9.

5.5 CONCLUSIONS

From the torque curves, it is evident that the above method gives a suboptimal solution. The actual minimum time solution depends on the separation between the knots also. In order to minimize time some of the knot spacings may need to be increased [13]. Here equal knot spacings have been considered.

CHAPTER 6

CONCLUSIONS

In this thesis two techniques of off-line optimal path planning have been presented Some of the salient features and limitations too are given below . Some suggestions for future scope in this area are also given .

61 SALIENT FEATURES AND LIMITATIONS

(1) A combination of B-Splines and MLV has been used for the optimal path planning of robot joint trajectories . The important feature of this is that MLV itself is efficient and simple to use , and use of B-Splines makes it even simpler and computationally more efficient .

(2) Spline interpolation for joint trajectories gives good approximation to the actual trajectory . Also , it ensures the continuity of the derivatives of the function to be interpolated ; which is desirable in robot path planning problems for meeting the constraints on them .

(3) In the Minimum Energy problem the total energy lost in the joint drive circuit has been minimized and hence the total electrical energy consumption . This is of significance in the case of robots which have to perform repetitive tasks . Most assembly like robots would fall into this category . Since motor

rating is dependent on temperature rise consideration and since temperature rise is dependent on the losses in the motor , this optimization enables proper choice of motor size . Alternately , for a given motor choice , the motors would be operated in the most efficient way to perform the stipulated task .

(4) Minimum Time control , that is , selection of a time optimal path between two specified EE positions , has been presented as a separate problem .

The procedures used for determination of optimal trajectories and the corresponding motor torques for the minimum energy and minimum time problems have been shown to be simple and computationally efficient .

(5) Dynamic and jerk constraints have been taken into account in both the problems , which is not the usual feature in most of the works done earlier .

(6) Both the methods can tackle obstacle constraints without any fundamental modification to the main algorithms .

If splines of high order are used , the simplicity of B-Spline approximation is reduced . The bandwidth of the banded matrix involved increases and the piecewise polynomial approximation tends towards becoming a pure polynomial approximation . High order B-Splines are not attractive from the view point of MLV also since the interval of local support of the splines increases as their order is increased . A judicious choice of spline order , is therefore necessary . We have found spline order of 5 to be the best for robot path planning using MLV .

6.2 SUGGESTIONS FOR FURTHER WORK

This work can be extended for path planning problems including obstacle constraints . If a good way of representing 3D objects in joint space is available , this method will prove to be efficient for collision -free path planning problems.

APPENDIX A

CALCULATION OF DYNAMICAL COEFFICIENTS [11]

The dynamical coefficients , that is , the elements of the matrix D , H and that of the vector G ,for the Lagrange Euler formulation of the first three joints of PUMA 560 (eqn 4.4) are calculated as follows.

$$\text{Let } S_i = \sin(q_i)$$

$$C_i = \cos(q_i)$$

$$[g_{i1} \quad g_{i2} \quad g_{i3}]^T = [m_i g\, \bar{x}_i \quad m_i g\, \bar{y}_i \quad m_i g\, \bar{z}_i]^T$$

where

$$g = 9\ 8062 \text{ m/sec}^2$$

Pseudo inertia matrix for ith joint is

$$J = \begin{bmatrix} J_{i\ 11} & J_{i\ 12} & J_{i\ 13} & J_{i\ 14} \\ J_{i\ 12} & J_{i\ 22} & J_{i\ 23} & J_{i\ 24} \\ J_{i\ 13} & J_{i\ 23} & J_{i\ 33} & J_{i\ 34} \\ J_{i\ 14} & J_{i\ 24} & J_{i\ 34} & J_{i\ 44} \end{bmatrix}$$

$$= \begin{bmatrix} A & m_i\bar{x}_i\bar{y}_i & m_i\bar{x}_i\bar{z}_i & m_i\bar{x}_i \\ m\ \bar{x}_i\bar{y}_i & B & m_i\bar{y}_i\bar{z}_i & m_i\bar{y}_i \\ m_i\bar{x}_i\bar{z}_i & m_i\bar{y}_i\bar{z}_i & C & m_i\bar{z}_i \\ m_i\bar{x}_i & m_i y_i & m_i z_i & m_i \end{bmatrix}$$

where

$$A = \frac{-I_{i\,xx} + I_{i\,yy} + I_{i\,zz}}{2} + m_i\,\bar{x}_i^{\,2}$$

$$B = \frac{I_{i\,xx} - I_{i\,yy} + I_{i\,zz}}{2} + m_i\,y_i^{\,2}$$

$$C = \frac{I_{i\,xx} + I_{i\,yy} - I_{i\,zz}}{2} + m_i\,\bar{z}_i^{\,2}$$

Now the coefficients are calculated as

$$D_{11} = J_{111} + J_{133} + J_{211}\,C_2^2 + 2\,J_{214}\,a_2\,C_2^2 + J_{222}\,S_2^2 + J_{233}$$

$$+ J_{244}(a_2^2\,C_2^2 + d_2^2) + J_{311}(S_3S_2 - C_3C_2)^2 + J_{322}$$

$$+ J_{333}(S_3C_2 + C_3S_2)^2 + J_{344}(a_2^2\,C_2^2 + d_2^2)$$

$$+ 2\,J_{334}\,(a_2\,S_3\,C_2^2 + a_2\,C_3\,S_2C_2)$$

$$D_{22} = J_{211} + J_{222} + 2\,J_{214}\,a_2 + J_{244}\,a_2^2 + J_{311} + J_{333} + J_{344}\,a_2^2$$

$$+ 2\,J_{334}\,a_2\,S_3$$

$$D_{33} = J_{311} + J_{333}$$

$$D_{12} = D_{21} = J_{214}\,d_2\,S_2 + J_{244}\,a_2\,d_2\,S_2 + J_{344}\,a_2\,d_2\,S_2$$

$$+ J_{334}\,d_2\,(S_3S_2 - C_3\,C_2)$$

$$D_{13} = D_{31} = J_{334}\,d_2\,(S_3S_2 - C_3C_2)$$

$$D_{23} = D_{32} = J_{331} + J_{333} + J_{334}\,a_2\,S_3$$

$$H_{111} = 0$$

$$H_{122} = J_{214}\,d_2\,C_2 + J_{244}\,a_2\,d_2\,C_2 + J_{344}\,a_2\,d_2\,C_2$$

$$+ J_{334}\ d_2(S_3C_2 + C_3S_2)$$

$$H_{133} = J_{334}\ d_2\ (S_3C_2 + C_3S_2)$$

$$H_{211} = J_{211}\ S_2\ C_2 - J_{222}\ S_2\ C_2 + 2\ J_{214}\ a_2\ S_2\ C_2 + J_{244}\ a_2^2\ S_2\ C_2$$
$$+ (J_{333} - J_{311})(S_3^2\ S_2C_2 + S_3C_3S_2^2 - S_3C_3C_2^2 - C_3^2\ S_2C_2)$$
$$+ J_{344}\ a_2^2\ S_2C_2 + J_{334}(2\ a_2S_3S_2C_2 + a_2C_3S_2{}^2 - a_2C_3C_2{}^2)$$

$$H_{222} = 0$$

$$H_{233} = J_{334}\ a_2\ C_3$$

$$H_{311} = (J_{333} - J_{311})\ (S_3^2\ S_2\ C_2 + S_3\ C_3\ S_2^2 - S_3C_3C_2^2 - C_3^2\ S_2\ C_2)$$
$$+ J_{334}\ (a_2S_3S_2C_2 - a_2C_3C_2^2)$$

$$H_{322} = -J_{334}\ a_2\ C_3$$

$$H_{333} = 0 \qquad H_{123} = H_{133}$$

$$H_{112} = -H_{211} \qquad H_{212} = 0$$

$$H_{113} = -H_{311} \qquad H_{213} = 0$$

$$H_{223} = -H_{322} \qquad H_{312} = 0$$

$$H_{313} = 0 \qquad H_{323} = 0$$

$$G_1 = 0$$

$$G_2 = g_{33}\ (S_3C_2 + C_3S_2) + C_2g_{21}$$

$$G_3 = -g_{33}\ (S_3C_2 + C_3S_2)$$

APPENDIX B

LINK PARAMETERS AND MASS PROPERTIES OF PUMA 560 ROBOT

The link parameters associated with the first three joints of the PUMA 560 robot arm are given below .

Joint No. (i)	α_i (degrees)	a_i (meters)	d_i (meters)
1	-90	0	0
2	0	0.432	0.1495
3	90	0	0

The mass properties are specified as follows :

I_i , the inertia matrix for ith joint is given below .

$$I_i = \begin{bmatrix} I_{i\,xx} & 0 & 0 \\ 0 & I_{i\,yy} & 0 \\ 0 & 0 & I_{i\,zz} \end{bmatrix}$$

$$\text{diag. } I_1 = [0.0071, \ 0.0267 \ , \ 0.0267] \text{ Kg-meter}^2$$

$$\text{diag. } I_2 = [0.1000, \ 0.7300, \ 0.8025] \text{ Kg-meter}^2$$

$$\text{diag. } I_3 = [0.0222, \ 0.2160, \ 0.2245] \text{ Kg-meter}^2$$

$\bar{S}_i = [\ \bar{x}_i \ , \bar{y}_i \ , \bar{z}_i \ ]$ is the position vector of the centre of mass of link i with respect to the ith coordinate system .

$$\bar{S}_1 = [0.0, \ 0.0 \ , \ 0.073]^T \text{ meter}$$

$$\bar{S}_2 = [-0.432, \ 0.0, \ 0.0]^T \text{ meter}$$

$$\bar{S}_3 = [0.00, \ 0.00, \ 0.10]^T \text{ meter}$$

m_i is the mass of ith joint .

REFERENCES

[1] C. de Boor, *A Practical Guide To Splines*, New York : Springer-Verlag, 1978

[2] M. J. D. Powell, *Approximation Theory and Methods*, London : Cambridge Univ. Press, 1981.

[3] J. Y. S. Luh and C. S. Lin , *Optimal path planning for mechanical manipulators*, ASME J. Dynam Syst. , Measurement, Contr. , Vol. 2, pp. 330-335 , June, 1981.

[4] K. G. Shin and N. D. McKay, *Minimum-Time control of robotic manipulators with geometric path constraint*, IEEE Trans. Automatic Contr. , Vol. AC-30 , pp. 531-541 , June , 1985.

[5] J. E. Bobrow, S. Dubowsky and J. S. Gibson , *Time-Optimal control of robotic manipulators along specified paths*, Int. J. Robotics Res. , Vol. 4 , No. 3, pp. 3-17, Feb. 1985.

[6] Byung K. Kim and Kang G. Shin , *Minimum-Time path planning for robot arms and their dynamics*, IEEE Trans. Sys, Man, & Cyb., Vol. SMC-15 , No. 2, pp. 213-223, March/April 1985.

[7] Kang G. Shin and Neil D. McKay , *Selection of near-minimum time geometric paths for robotic manipulators*, IEEE Trans. on Automatic Control, Vol. AC-31, No. 6, pp. 501-511, June 1986.

[8] S. H. Suh and K. G. Shin, *A variational Dynamic Programming approach to robot-path planning with a Distance-Safety criterion* ,IEEE J. of Robotics and Automation, Vol. 4,No. 3, pp. 334-349, June 1988.

[9] H. H. Tan and R. B. Potts , *Minimum Time trajectory planner for the discrete dynamic robot model with dynamic constraints*, IEEE J. of Robotics and Automation, Vol. 4, No. 2, pp. 174-185, April 1988.

[10] James E. Bobrow, *Optimal robot path planning using the minimum time criterion*, IEEE J. of Robotics and Automation, Vol. 4 , No. 4, pp. 443-450, August 1988.

[11] Patrikar A. M , *Optimal path planning of robot manipulators by the method of local variations*, M. Tech. thesis, Dept. of Electrical Engg. , IIT Kanpur, Dec. 1987.

[12] Seshadri Commuri, *Optimal Collision-free path planning*, M. Tech. thesis, Dept. of Electrical Engg. , IIT Kanpur, Feb. 1989.

[13] C. S. Lin, P. R. Chang and J. Y. S. Luh, *Formulation and Optimization of cubic polynomial joint trajectories for industrial robots*, IEEE Trans. Automat. Contr., Vol. AC-28, pp. 1066-1074, Dec. 1983

[14] Stuart E. Thompson and Rajnikant V. Patel, *Formulation of joint trajectories for industrial robots using B-Splines*, IEEE Trans. Industrial Electronics, Vol. IE -34, NO. 2, pp. 192-199, May 1987.

[15] Ghosh A. and Patrikar A. M., *Optimal path planning using the method of local variations*, paper no. MS-89-280, Proc SME world conference on Robotic Research Maryland 1989. Also to appear in SME Trans. on Robotics and Automation.

[16] Ghosh A., Seshadri C. , & Patrikar A. M., *Time Optimal path planning for robot manipulators*, to appear in the proceeding of IEEE Int. Symposium on Ckts. & Systems. New Orleans. MAy 1990. Paper No. Rb # 3.

[17] Fedorenko R. P., *The basis of the method of variations in phase space for the numerical solution of optimal control problems*, Zh. vychisl. Mat. Fiz ; 9, 6, 1396-1402, Sep. 1969

[18] F. L. Chernous'Ko, *A local variation method for the numerical solution of variational problems* , Zh. vychisl. Mat. mat. Fiz 5, 4, pp. 749-754, 1965.

[19] I. A. Krylov and F. L. Chernous'Ko , *Solution of problems of optimal control by the method of local variations*, Zh. vychisl. mat. Fiz. 6, 2, 203-217, 1966.

[20] Podval'Nyi L. D., *A numerical method for solving optimal control problems*, Zh. vychisl. Mat. Fiz. 9, 2, pp.300-314, May 1969.

[21] Banichuck N. V. , Petrov V. M. and Chernous'Ko F. L. , *The method of local variations for variational problems involving non-additive functionals* Zh. vychist Mat. mat. Fiz. 9, 3, pp. 548-557, October,1969.

[22] Vatel I. A. and Kononenko A. F. , *A numerical scheme for solving optimal control problems*, Zh. vychist, Mat. mat. Fiz. 10, 1, pp. 67-73, 1970.

EE-1990-M-SIN-OPT